PROJECT SCHEDULING THROUGH ZERO-ONE PROGRAMMING

By

ATUL P. BAHADUR

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

MAY, 1975

PROJECT SCHEDULING THROUGH ZERO-ONE PROGRAMMING

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

By
ATUL P. BAHADUR

to the

DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
MAY, 1975

ME-1975-M-BAH-PRO

CERTIFICATE

This is to certify that the thesis entitled 'Project Scheduling Through Zero-One Programming' by Atul Prakash Bahadur has been carried out under my supervision and has not been submitted elsewhere for the award of a degree.

[illegible]
(J.L. BATRA)
Assistant Professor
Department of Mechanical Engg.
Indian Institute of Technology
KANPUR

ACKNOWLEDGEMENTS

I take this opportunity to express my gratitude to Dr. J.L. Batra for his invaluable guidance, keen interest and constant encouragement throughout the course of this work.

My thanks are also due to Mr. R.K. Patil for the assistance rendered in the compilation of the work.

Lastly, I thank Mr. D.P. Saini for the efficient typing of the manuscript.

A.P. BAHADUR

TABLE OF CONTENTS

SYNOPSIS

of the
Dissertation
on
'Project Scheduling through Zero-One Programming'
Submitted in Partial Fulfilment of the
Requirement for the Degree
of
MASTER OF TECHNOLOGY IN MECHANICAL ENGINEERING

by
A.P. BAHADUR

Department of Mechanical Engineering
Indian Institute of Technology, Kanpur
May, 1975

✓ The present work deals with the project scheduling problems. Here an attempt has been made to develop a zero-one formulation which can accommodate a variety of practical situations including pre-emption, concurrency and non-concurrency, force-finishing and variation of resource requirements during the execution of the activities. The objective functions considered are minimization of make-span, throughput time and penalty for late completion. ✓ (Bàlas' and 'Resource - Time - Varying' algorithms (a variation of Balas') have been used to solve the formulated problems. The results indicate that this formulation has an edge over Bowman's, and Moodie and Mandeville's and is comparable with the Fischer's and Davis' models.)

CHAPTER 1

INTRODUCTION

Project Planning models based on network structure first appeared in the year 1958 under the acronym PERT (Program Evaluation and Review Technique) and CPM (Critical Path Method). These models have been further developed and refined and have been used in a variety of projects, including building constructions, publishing ventures, research projects, defence works and industrial management. Both PERT and CPM assume the availability of unlimited resources which in many real problems is not a valid assumption. Hence during the last decade various methods have been developed for dealing with the case of limited resources.

The resources of any organization can be categorised into five major classes. These are - manpower, machine, money material and (often overlooked) time. It is the interaction of the three main variables of time, money and manpower which adds intricacies to the already complex problem. We can accelerate an activity by using more manpower, but the excess availability of material may not be of any help. Under certain circumstances, we may have to split, force-finish or link some activities. Apart from the above constraints we may have indirect limitations that are beyond the control of planners but they should, at least, receive some considerations. These are: weather (it affects crew efficiency, material procurement, labour turnover), space (limited working space determines the crew size), efficiency (some crews and pieces of equipment are more efficient than others) and reliability (critical jobs should be given to more reliable resources).

The constrained resource problem has been categorized broadly into the following two types:

1. Maximization of some function of resource utilization subject to a project duration. In this case we can have minimization of maximum demand or the minimization of the sum of the squared/absolute deviation from the mean.

2. Minimization of the project duration subject to a given resource availability. For multiproject problems we can minimize some penalty function associated with the non-completion of the projects.

The reasons for the non-availability of the generalized analytical solution procedures for the constrained resource problems are because of the difficulties encountered in making a mathematical formulation. Some of the difficulties are listed below:

1. Generally the activities can be started or continued with a reduced level of resources. This, along with the substitution of resources, greatly complicates the problem of various alternatives.

2. In a network consisting of the activities of the various departments it may be desirable that we do the resource levelling for each department rather, than the whole network. Such problems are too complex for most of the existing methodologies.

3. In most of the solution procedures we find that either the model grows too large for any practical sized problem or the solution procedure becomes very lengthy.

4. The feasible solutions achieved from a solution procedure may be quite off compared to the optimal solution.

5. The interdependence of the activities as a result of sharing the same resource poses serious problems for the mathematical formulation.

6. In case of multiproject problems different projects may have different priorities and this should be taken into consideration for project scheduling.

7. A number of network problems may require activity splitting or concurrency of activities and this has to be mathematically expressed.

Keeping the above problems into consideration, an attempt has been made to develop a zero-one formulation. In this work the emphasis is on two main points:

1. The model should be able to solve a wide range of network problems.

2. The model should possess some special structure which may be exploited for solving any practical sized problem.

A brief survey of the literature is presented in Chapter II. A generalized problem formulation is given in Chapter III. In Chapter IV, the solutions obtained using the proposed methodology and the discussions of the results are presented.

CHAPTER 2

LITERATURE SURVEY

The current approaches to solve the resource constrained network problems can be categorized broadly under the following headings:

1. Heuristic methods.
2. Combinatorial procedures:
   a. Modification of feasible schedule
   b. Construction of a schedule by means of a sequence of partial schedules.
3. Mathematical programming methods.

A brief description of these methods is given in the following paragraphs:

2.1. Heuristic Methods:

MAP[1] which stands for Multiple Resource Allocation Procedure is a heuristic technique that considers the constraints of resource availability, dead lines, activity relationship etc. to determine the near optimal schedule. It is based on a set of eight rules which determine the crew size and the length of the project duration. Priorities are assigned to the jobs that have the same starting time according to the following rules:

a) least total float,

b) larger need for overall resource requirement,

c) large crew size and

d) sequence code.

The main drawback of MAP lies in its inability to do resource levelling. In case of multiprojects it attaches the same priority to each project.

J.D. Wiest's(2) model SPAR-1 (Scheduling Program for Allocation of Resources) handles multiple projects with fixed and variable crew sizes. It can handle jobs requiring multiple resources like machine, money, each of which are limited in quantity. Two cost functions used by Wiest are:

1. Maintaining the crews at peak loads and paying them irrespective of whether they are active or not.

2. Assigning exponential increase in penalties when the project is not completed within the due dates.

This model has been used in quite a number of large projects. It generates feasible solutions which are sometimes quite off the optimal one. The author has cited an example of this in the case of a project which involved sixteen activities. It was found that the priority rule based on total slack resulted in a non-optimal solution.

In another paper Wiest(3) has proposed an extension of his original model. In this model (called SPAR-2) instead of defining a critical path, he defines critical sequence which is determined by technological orderings set of job times and resource constraints. Moreover, the critical sequence depends upon a given feasible schedule. He has discussed several properties of the large projects. The work is extended to the cases where resource availability is varying during the scheduling period. However, the author has not solved any practical sized problem to confirm the utility of the technique.

Burgess and Killebrew(4) have solved the resource allocation problem for the repetitive network. They have used the method of minimizing the sum of squares of the resource requirements on various days. By moving the activities backward and forward (keeping in view of the technical constraints) the minimum sum is achieved. Various suggestions have been proposed by the author to achieve this minimum. However, none of these approaches guarantee convergence to optimality. Dewitte(7) uses the Burgess and Kil$\underset{\wedge}{\overset{l}{}}$brew's routine with different measure of effectiveness; viz, the minimization of the sum of absolute magnitude of fluctuation from a calculated mean level of usage.

Levy, Thompson and Wiest[8] have employed the Critical Path Method of analysing the job to be done on a given project. Jobs are assigned at their earliest possible start time and the resource profile is drawn for each project. Then the program sets the 'trigger levels' one unit below the peak requirements in each of the projects and attempts to reschedule the activities. They have taken an example of 55 activity network distributed in 4 shops. The authors claim that their procedure can be used for a 500 activity network. The effectiveness of the procedure depends upon the the amount of slack in the activities and the number of non-critical activities.

Wilson[5] has incorporated Dynamic Programming to the Levy's method. However, it suffers from a major drawback that each unit is assumed to have one resource unit only. Kelley[9] has developed serial and parallel methods of solving the resource allocation problem. In the former, he determines the early start time of each activity and attempts to schedule it at that time. If the required resources are not available the activities are delayed to the earliest feasible start time at which resources are available. Activities can be split arbitrarily into integral time units. Unlike serial methods which schedule one activity at a time, parallel methods schedule several activities at a time. The procedure starts with the determination of the set of activities P(t), which are in process or could start at time t and the subset of activities Q(t) in P(t) which can be scheduled at time t. The author claims that serial methods are more practical than the parallel methods.

RAMPS(10), an acronym for Resource Allocation and Multiproject Scheduling, is a computerized method for handling several projects subject to some specified constraints. Three sets of input data are required for each activity; amount of resource required, time required and cost of splitting the activity once it has begun. Further more, starting dates of the projects, desired completion dates, penalty rates and project priorities are required. Since details of 'RAMPS' computational algorithm are not available, it is not possible to analyse the method. However it is claimed that it can solve network comprising of 700 activities with 20 different types of resources.

James H. Patterson and Walter D. Huber(30) have suggested minimum bounding/maximum bounding and search algorithms to determine the duration of the project. They have compared their results with those of Davis(19) and Moodie and Mandeville(23) and they claim that their results are favourable.

2.2 Combinatorial Procedures

a) Schedule Modifying Methods:

In these methods we start with a schedule that satisfies the precedence constraints and then we modify the schedules so as to obtain an optimal solution which satisfies resource requirements. In a CPM network the longest path corresponds to the duration of the network and is computed by a simple recursive relationship. Similar attempts have been made for the formulation of an equivalent problem of finding a minimaximal path in a disjunctive graph.

Bennington & Wennis(11) have considered the case of two machine sequencing problems in which they attach pseudo precedence relationships for the activities requiring the same machine. The disjunctive graphs so obtained cannot be solved by CPM because the longest path in this case is unbounded. These problems of machine sequencing are referred to as the two-state minimaximal path problem. The project scheduling problem is slightly more complicated because there may be more than one unit of each resource available and several activities may overlap in the feasible solution. This problem is referred as a three state minimaximal path problem in a disjunctive graph.

Florian, Trepant and McMohan(12) have suggested a method to solve the two state minimaximal problem of machine sequencing. Their algorithm for constructing the tree of resolution is quite effective. They first attempt to find a feasible solution, then attempt to modify it and then try to prove the optimality. They claim that certain modifications can be incorporated to take into account the setup time for each operation. They have solved upto 10 machines - 100 jobs sequencing problems. They claim that the solutions obtained by their methods are comparable to the solutions obtained by the solution procedures of Schrage's (13) and Balas'(14).

Sussman(16) has made an improvement over Florian, Trepant and McMohan's(12) procedure by taking the partial resolutions at the earlier conflicts and finding the bounds of the resolved graphs. Balas'(14) method of finding the resolution consists of starting with any feasible sequence. Each new term of the sequence is generated from an earlier one by complementing one disjunctive arc. Here the search is drastically cut down because the disjunctive arcs that are to be considered for being complemented fall over the critical path. The bounds are calculated at each stage to determine the search direction. The author has not given any example but he claims that his method can generate a reasonably 'good' feasible solution.

There is only one method, which was proposed by Gorenstein(15), for solving the project scheduling problem by schedule modification. He finds that the main problem comes in determining the generalized coefficient of stability GCS_j for each resource j. These GCS_js are used not for bound determination but for identifying the resource feasibility of the resolution. Further more, if GCS_j is less than resource availability it implies resource feasibility, but the converse is not necessarily true. Hence it may be possible that the optimal solution is not achieved in using the GCS_j criterion.

b) Schedule Construction Methods:

As the name implies, the general approach in these algorithms is to 'build up' a feasible solution. In almost all of the following methods an ordering or permutation of the jobs is tried to build up the schedules. Schrage(13) has evolved a branch and bound method for implicitly enumerating all schedules and determining the optimum. He has considered the following constraints: (1) Precedence (2) Resource availability (3) Non-preemption.

The procedure suggested is essentially a tree generating procedure. For each partial schedule or branch in the tree one can calculate lower bounds for all complete schedules that can be generated on that time. For obtaining the bounds on the tree he suggests (1) Precedence based bounds, or, (2) Resource based bounds. The objective function is min. (max. finish time) . The generalizations are applicable to the following cases (a) multiresources (b) varying resource availability (c) 'OR' activities (d) unavailability of resources for changeover between activities (e) wide range of objective functions.

Schrage(17) has suggested improvements for taking preemptive cases. For the objective of min.(max. finish time) the bounds are either precedence or resource based. He indicates that the preemptive version (a) requires slightly more computation (b) does not result in a substantially lower min-max completion time. Mason and Moodie's(18) method minimizes the combined cost of fluctuations in resource demand and delay of project duration. Resource requirement and duration of each activity is represented by a vector. At each time the amount of work remaining and the total cost incurred are determined. These costs are used for further branching of the tree. The resource availability constraints can be used at a node to eliminate some subsets of the tree. According to the authors, the results indicate that the optimal schedule is relatively insensitive to changes in the ratio of project delay cost and the resource fluctuation cost.

Davis and Heidorn's$^{(19)}$ algorithm permits the computation of optimum project duration under the conditions of multiple resource requirement per job. The sample network is converted into unit duration activity network by imposing the constraints 'must immediately precede' along with 'normal precedence'. The number of stages are equal to the number of time periods in the longest path. Feasible subsets are generated and from them 'A network' of the sample problem is constructed. Direct arcs are drawn between nodes if precedence and resource constraints are met. The trouble arises due to the fact that if activity durations are large the number of feasible solutions shall be extremely large and hence the identification of 'A-network' will be problematic. The authors have suggested the use of dynamic programming approach for making 'A-network' from the feasible subsets. Branching is simplified by using, (1) precedence based or (2) resource based criteria. This algorithm can also account for the variations in job continuity and varying resource requirements.

Johnson(20) has formulated branch & bound algorithm for the construction of the schedule. Johnson's algorithm uses the following logic for the construction of the schedule:

1. Given a partial schedule S_k at time t_k, consider the set of all unschedule activities whose immediate predecessor have been scheduled.

2. From (1) form subsets of activities so that resource constraints are considered.

3. By including these subsets form new subsets.

He considers two bounds for each partial schedule, (a) the critical path gives one bound and, (b) resource requirement of the other. Based on computational experience, Johnson claims that his is the most comprehensive algorithm for scheduling single resource problems.

2.3 Mathematical Programming Methods:

H.H. Greenberg(22) has developed mixed integer formulation for the general n job - m machine sequencing problem. He has used S_{ip} and d_{ip} to denote the start time and duration of job i on machine p. Further he defines:-

$$X_{ij}^{p} = \begin{cases} 1, & \text{if job i starts on machine p before job j.} \\ 0, & \text{otherwise.} \end{cases}$$

With these he has specified the technical constraints and machine availability constraints. By dropping out the integrality restrictions the problem is solved by branch and bound. It is worth noting that the linear program solved at each node is nothing more than a CPM problem for which we have better methods in disjunctive graph technique.

Bowman(24) has given the zero-one formulation of schedule sequencing problem. He has considered the example of 3 products which have to be processed on 4 machine in a given technological ordering. He uses $X_{A:i} = 1$ to denote that a product x is being processed on machine A at time i and $X_{A:i} = 0$ otherwise. A large penalty is attached to the jobs being processed at the end and this helps in finding the shorter make span of the project. The number of variables used in his method are extremely large (are equal to (products) x (machines) x (time periods)). The details of this method are given in appendix A.

Manne(25) has suggested an improvement over this method. Integer valued unknowns X_j's are used to indicate the day on which task j is to be started. He has converted the noninterference constraints into the equivalent constraints by introducing the zero-one variable Y_{jk} for jobs j and k. These modifications are used for expressing the sequencing constraints, due date requirement, and exact delays between two consecutive jobs. The number of variables required in Manne's method are considerably less than in Bowman's(24) method. Manne has suggested that all attempts should be made to reduce Y_{jk}'s because these are required only in noninterference constraints and in any numerical problem these classes may occur only in a few cases. In Wagner's(26) method, the number of constraints to be satisfied are staggering but most of them are inoperative. He claims (without any proof) that his model can solve a wide class of machine sequencing problems.

Moodie and Mandeville(23) have solved the resource-balancing problem by using the Bowman's(27) method for solving the assembly line balancing problem. The difference between the two class of problems is that an activity in assembly line balancing problem occurs at one and only one station while, in resource balancing problem, an activity may spread over several days. An Integer Linear Programming formulation is used to minimize the cost of applying the resources. Even the multiproject-resource-balancing problems have been solved by these methods. The details of the formulation of Moodie and Mandeville(23) are given in Appendix B.

Fischer(28) has used Lagrange multipliers to obtain a strong lower bound on the cost of an optimal solution for the resource constrained network scheduling problems. A branch and bound procedure is used to obtain the lower bounds. The concept of active schedule has been used to eliminate nodes in the tree and to update multipliers at each branching. The author has suggested that the efficiency can be improved by the elimination of inactive schedul$\overset{es}{\text{ing}}$. He assumes that one type of resource is available at each time period. The details of his method are given in Appendix C.

CHAPTER 3

PROBLEM FORMULATION AND SOLUTION METHODOLOGY

The generalized problem formulation of a project scheduling problem is presented in this Chapter.

3.1 General Statement of the Problem:

'For a project scheduling problem, minimize either throughput time or make-span or penalty for late completion subject to the various constraints. The various constraints considered account for the concurrency and non-concurrency of activities, availability of resources, technological ordering, preemption, force-finishing of activities, etc.'

3.2 Nomenclature:

The following nomenclature is used to define the problem:

i : project number ; $i = 1,2,\dots, I$.

j : activity number; $j = 1,2,\dots, N_i$;

N_i = number of activities in project i.

t : time period.

T : period for completion of all the projects.

D_{ij} : duration time of activity j of project i.

EA_{ij} : earliest finish time of activity j of project i.

LA_{ij} : latest finish time of activity j of project i.

EP_i : earliest finish time of project i.

LP_i : latest finish time of project i.

$R_{ijkt'}$*: amount of resource of type k required by activity j of project i at time t'. Here t' is the time since the activity has begun.

k : resource number, $k = 1, \ldots, K$; where K is the number of different types of resources required.

RA_{kt} : amount of resource of type k available at time t.

T_{ic} : cycle time after which the project i is repeated.

X_{ijt} : avariable which takes the value 0 or 1; $X_{ijt} = 1$ indicates that the jth activity of project i is completed in time t and X_{ijt} is zero otherwise. X_{ijt} is a variable only from EA_{ij} to LA_{ij} because it can be assigned a value of zero for $t < EA_{ij}$ and $t > LA_{ij}$.

* If the resource requirement of an activity does not vary during its duration, we shall use only R_{ijk}.

Y_{it}: a variable which takes the value 0 or 1; $Y_{it} = 1$ indicates that the project i is completed by time $(t - 1)$ and zero otherwise. Y_{it} is a variable from (EP_i+1) to LP_i only because it can be assigned the value zero for $t \leq EP_i$ and one for $t > LP_i$.

Z_t: a variable which takes the value 0 or 1; $Z_t=1$ indicates that all projects have been completed by time $(t - 1)$ and zero otherwise. It is a variable for $t > \max EP_i$ to $t \leq T$.

P_{it}: penalty incurred due to non-completion of project i in time t.

3.3 Assumptions:

The following assumptions have been made for the mathematical formulation of the project scheduling problems:

1. The time duration of each activity is deterministic.
2. No crashing of activities is allowed.
3. Resource requirement of each activity is known.
4. Resourse availability throughout the project duration is known.

5. Duration time of each activity is independent of the other activity.

6. No alternate sequencing of an activity is permitted.

7. All activities in a project are equally important.

8. Resource substitution of an activity is not allowed.

3.4 Constraints:

The various constraints can be expressed mathematically as follows:

1. Activity Completion Constraints

Each activity of the project has to be completed. Mathematically this can be represented as,

$$\sum_{t=EA_{ij}}^{LA_{ij}} X_{ijt} = 1 \qquad (1)$$

Then, by definition

$$X_{ij,LA_{ij}} = 1 - \sum_{t=EA_{ij}}^{LA_{ij}-1} X_{ijt} \qquad (2)$$

Since X_{ijt} can be either 0 or 1, we can write Eq.(1) in an equivalent form as given below:

$$\sum_{t=EA_{ij}}^{LA_{ij}-1} X_{ijt} \leqslant 1 \qquad (3)$$

The formulation as given in Eq. (3) has the advantage that the number of variables in an activity is reduced by 1.

2. Project Completion Constraints:

For the completion of project i by time (t - 1) we must complete all its activities in time (t - 1). Mathematically,

$$N_i\, Y_{it} \leqslant \sum_{j=1}^{N_i} \sum_{p=EA_{ij}}^{t-1} X_{ijp} \qquad i = 1, \ldots, I\ ;$$

$$\begin{cases} t > EP_i \\ t \leqslant T \end{cases} \qquad (4)$$

If all the projects have to be completed by time (t - 1), we require,

$$Z_t \sum_{t=1}^{I} N_i \leqslant \sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{p=EA_{ij}}^{\substack{t-1 \\ \cancel{LA_{ij}}}} X_{ijp}$$

$$t > \max EP_i\,.$$

$$t \leqslant T \qquad (5)$$

3. Force-Finish Constraints:

An activity ij is said to be force-finished if it has to be force finished by time t_f ($EA_{ij} \leq t_f \leq LA_{ij}$). To satisfy this constraint we modify Eq. (1) as follows:

$$\sum_{t=EA_{ij}}^{t_f} X_{ijt} = 1 \qquad \ldots \qquad (6)$$

4. Sequencing Constraints:

Let activities j_1 and j_2 be the two activities of the project i. If j_1 precedes j_2, this can be represented by the following inequality:

$$\sum_{t=EA_{ij_1}}^{LA_{ij_1}} t\, X_{ij_1t} + D_{ij_2} \leq \sum_{t=EA_{ij_2}}^{LA_{ij_2}} t\, X_{ij_2t} \qquad (7)$$

The precedence relationships between the activities of the different projects can be expressed as,

$$\sum_{t=EA_{ij}}^{LA_{ij}} t\, X_{i_1j_1t} + D_{i_2j_2} \leq \sum_{t=EA_{i_2j_2}}^{LA_{i_2j_2}} t\, X_{i_2j_2t} \qquad (8)$$

where i_1 and i_2 refer to different projects. Furthermore, if we want to have some lag (say ν)

between the two consecutive activities we can express it by imposing the equality constraint. Mathematically this is expressed as,

$$\sum_{t=EA_{i_1j_1}}^{LA_{i_1j_1}} t\, X_{i_1j_1t} + D_{i_2j_2} + \gamma = \sum_{t=EA_{i_2j_2}}^{LA_{i_2j_2}} t\, X_{i_2j_2t} \qquad (9)$$

5. Concurrency and Non-Concurrency Constraints:

Two activities are said to be concurrent if both of them have to be carriedout simaltaneously. If either of them has to be completed before the other starts, we call them as non-concurrent activities. To ensure that an activity ij is being carriedout at time t we must satisfy the following relationship.

$$\sum_{p=t}^{t+D_{ij}-1} X_{ijp} = 1 \quad \ldots\ldots \qquad (10)$$

Hence if we want that only one of the activities i_1j_1 and i_2j_2 be carriedout at time t, we should satisfy the following inequality.

$$\sum_{p=t}^{t+D_{i_1j_1}-1} X_{i_1j_1p} + \sum_{p=t}^{t+D_{i_2j_2}-1} X_{i_2j_2p} \leq 1 \qquad (11)$$

For non-concurrent activities Eq. (11) should be satisfied for the following range of t.

$$\text{Min.}(LA_{i_1j_1}, LA_{i_2j_2}) \geqslant t \geqslant \text{Max.} \; ((EA_{i_1j_1} - D_{i_1j_1} + 1), (EA_{i_2j_2} - D_{i_2j_2} + 1))$$

For concurrency we need to add the resource requirement of both the activities. Mathematically it is expressed as given in Eq. (12).

6. <u>Preemption Constraints:</u>

Preemption of an activity implies that an activity can be interrupted, once it has started. Let us say that an activity ij of duration D_{ij} is interrupted after duration D_{ij_1} and then restarted and completed without any further interruption. Thus the activity ij is divided into two subactivities ij_1 and ij_2 of durations D_{ij_1} and D_{ij_2} respectively, such that $D_{ij} = D_{ij_1} + D_{ij_2}$. On these two activities wecan easily impose the sequencing constraints as in (8). Similarly we can take into account more interruptions of an activity.

7. Resource Availability Constraints:

If the resource requirements of an activity do not vary during the execution period of the activity, the resource requirements at any time t would be given by,

$$\sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{p=t}^{t+D_{ij}-1} R_{ijk} X_{ijp} \leq RA_{kt} \quad t = 1,\ldots,T; \; k = 1,\ldots,K \qquad (12)$$

If the resource requirement of an activity changes during its execution period, we can satisfy the resource requirement at time t as follows:

$$\sum_{i=1}^{I} \sum_{j=1}^{N_i} \sum_{p=t}^{t+D_{ij}-1} R_{ijkt'} X_{ijp} \leq RA_{kt}, \quad t = 1,\ldots,T; \; k = 1,\ldots,K \qquad (13)$$

For the repetitive project the resource requirement has to be satisfied only for T_c time units.

3.5 Objective Functions:

The various objective functions considered in this study are concerned with the minimization of makespan, throughput time and penalty. Mathematically they can be expressed as follows:

1. Minimum Makespan:

The completion of all the projects in the minimum time can be expressed by the following objective function:

$$\text{Maximize} \quad Z = \sum_{\max.EP_i+1}^{T} Z_t \qquad (14)$$

2. Minimum Throughput-time:

The earliest completion of a project i can be achieved by the following objective function:

$$\text{Maximize} \quad Z = \sum_{t=EP_i+1}^{LP_i} Y_{it} \qquad (15)$$

The throughput-time of all projects can be minimized by the following relationship:

$$\text{Maximize} \quad Z = \sum_{i=1}^{I} \sum_{t=EP_i+1}^{LP_i} Y_{it} \qquad (16)$$

3. Minimum Penalty:

Let P_{it} be the penalty for the non-completion of project i in time t, then we can express the minimum penalty as follows:

$$\text{Maximize} \quad Z = \sum_{i=1}^{I} \sum_{t=EP_i+1}^{LP_i} P_{it} Y_{it} \qquad (17)$$

3.6 Solution Methodology:

Using the aforesaid generalized formulation a number of project scheduling problems have been formulated. The descriptions of these problems are presented in Chapter IV. The following algorithms have been used to solve the formulated problems.

1. Balas' algorithm for zero-one programming has been used to achieve the optimal solution. Since it takes many iterations to find the optimal solution, another algorithm is proposed.

2. Resource - Time - Varying algorithm, a variation of Balas' algorithm has been developed by which we generate the feasible solutions in such a way that they are optimal. The various steps required are given below:

   a) Schedule the projects based on the early start times and find the maximum resource requirement.

   b) Decrease the resource availability by one and find a feasible solution. If a solution is obtained, decrease the resource availability further by one and continue till we don't get a feasible solution.

c) Increase the completion time of the project in step of 1, till a feasible solution is obtained.

d) Repeat steps (b) to steps (d) until the resource availability is equal to the maximum resource requirement of an activity or when the resource utilization has reached a reasonable value.

The advantages of using the second algorithm are (1) we determine a schedule for each value of resource availability, (2) at each step we are finding a feasible solution (which is optimal) and this is computationally faster than finding the optimal from the feasible solutions, and (3) in step (b) we are doing the resource levelling.

CHAPTER 4

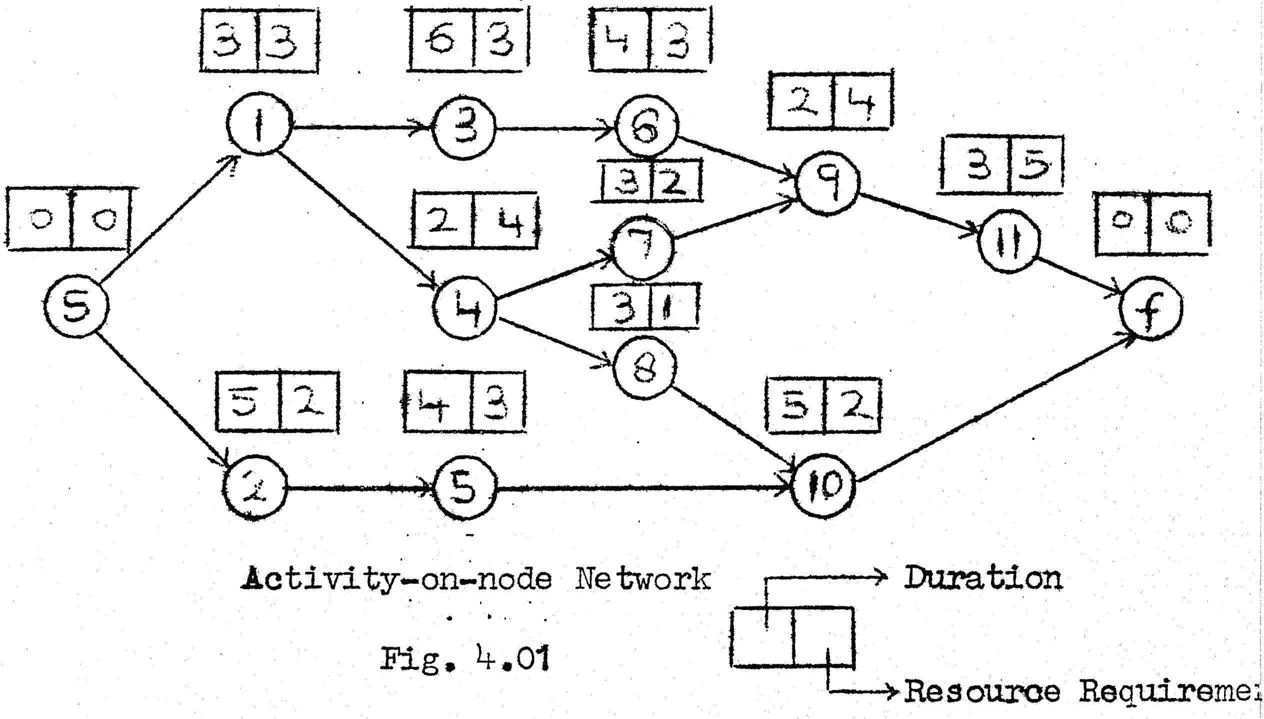
ILLUSTRATIVE PROBLEMS

In this chapter a few problems have been solved to illustrate the proposed formulation. The networks have been taken from various books and journals. For each problem, the network, the scheduled graph and the resource profiles have been developed.

Network 1:

An 11 activity, 1 resource, activity on node network is given in Fig. 4.01. The problem has been solved by the Resource-Time-Varying algorithm. The results are given below:

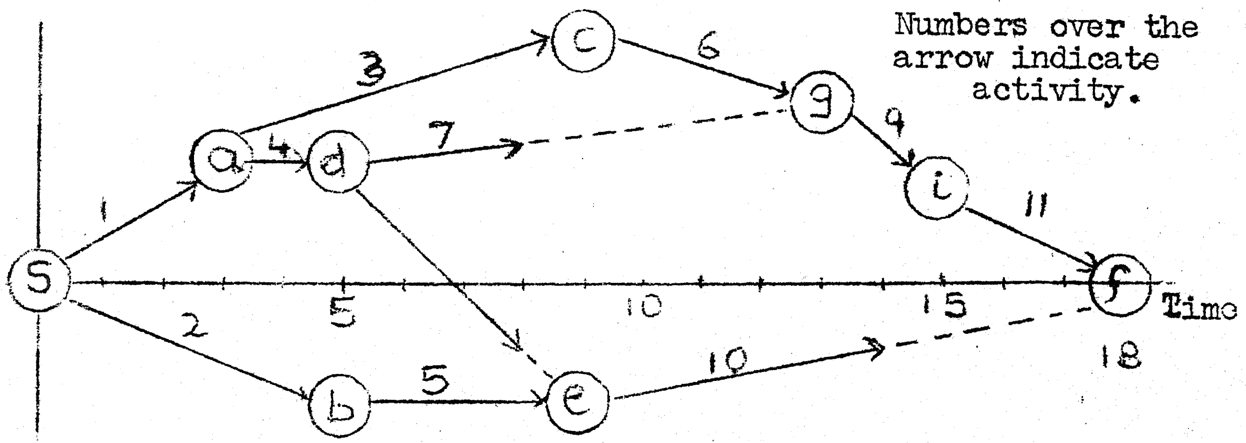
The scheduled graph and resource profile based on early start time are given in Fig. 4.02. The maximum resource requirement is 9. The resource availability is decreased to the level of 8 and a feasible solution is searched for T = 18. No solution exists for T = 18 and so we increase it to 19 and obtain a solution as shown in Fig. 4.03. For this schedule the maximum resource requirement comes out to be 7 only. Further, on decreasing the



Activity-on-node Network

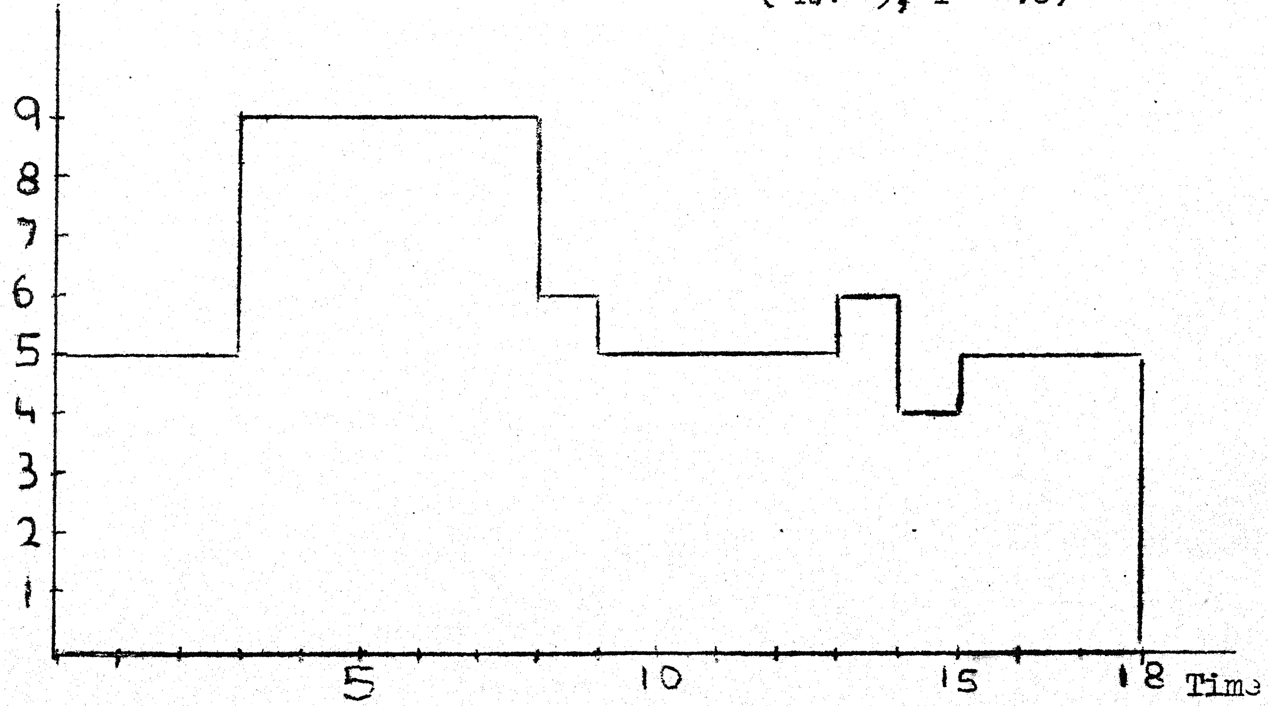
Fig. 4.01

resource availability to the level of 6 we get a feasible solution at T = 21. This is shown graphically in Fig. 4.04. The total number of iterations required for the solutions are 1270. The resource utilization for the three solutions are 111/162, 111/133 and 111/126 respectively.



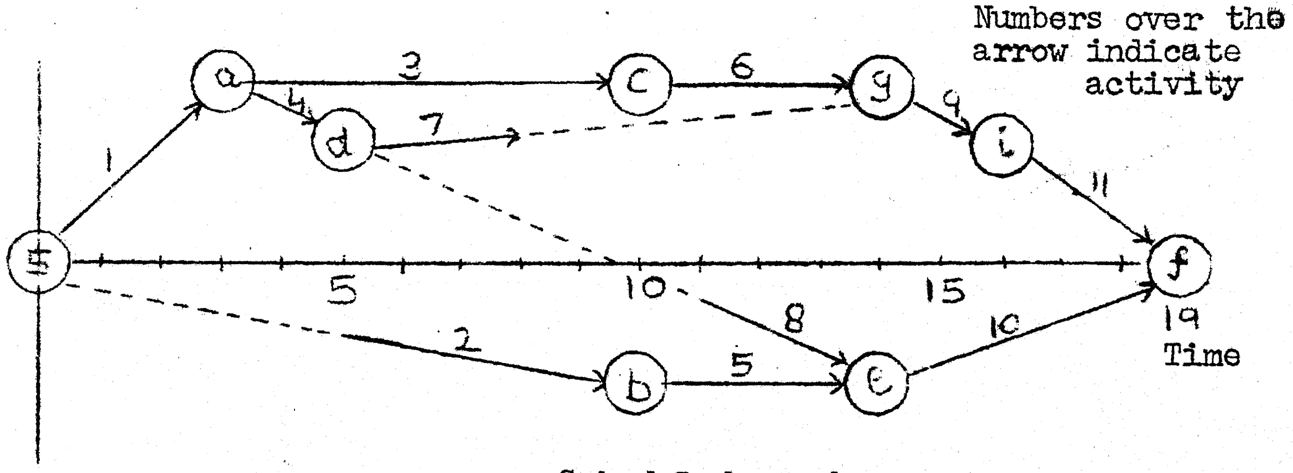
Scheduled graph

(RA= 9, T = 18)

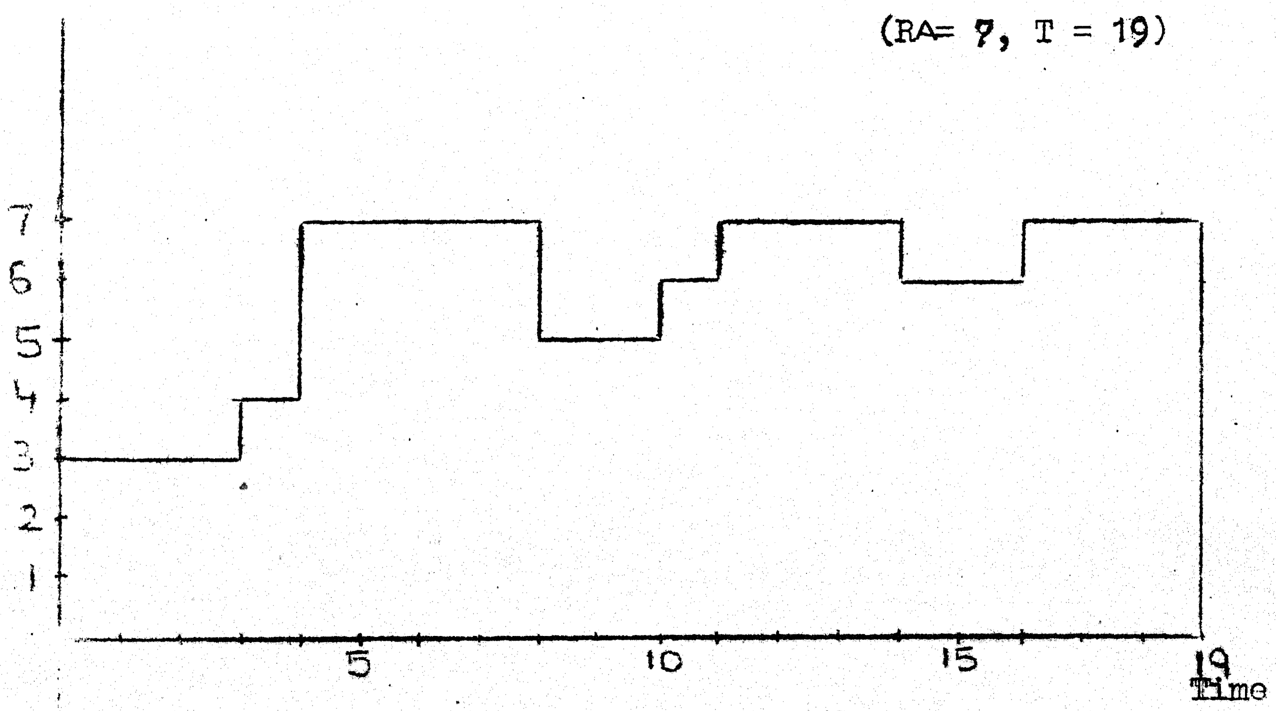


Resource Profile

Fig. 4.02

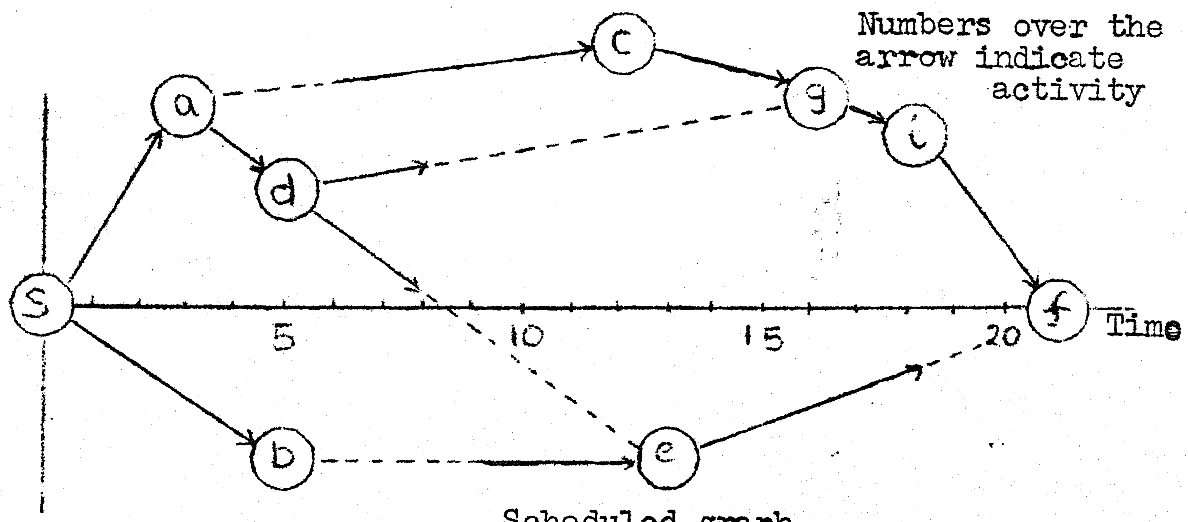


Scheduled graph

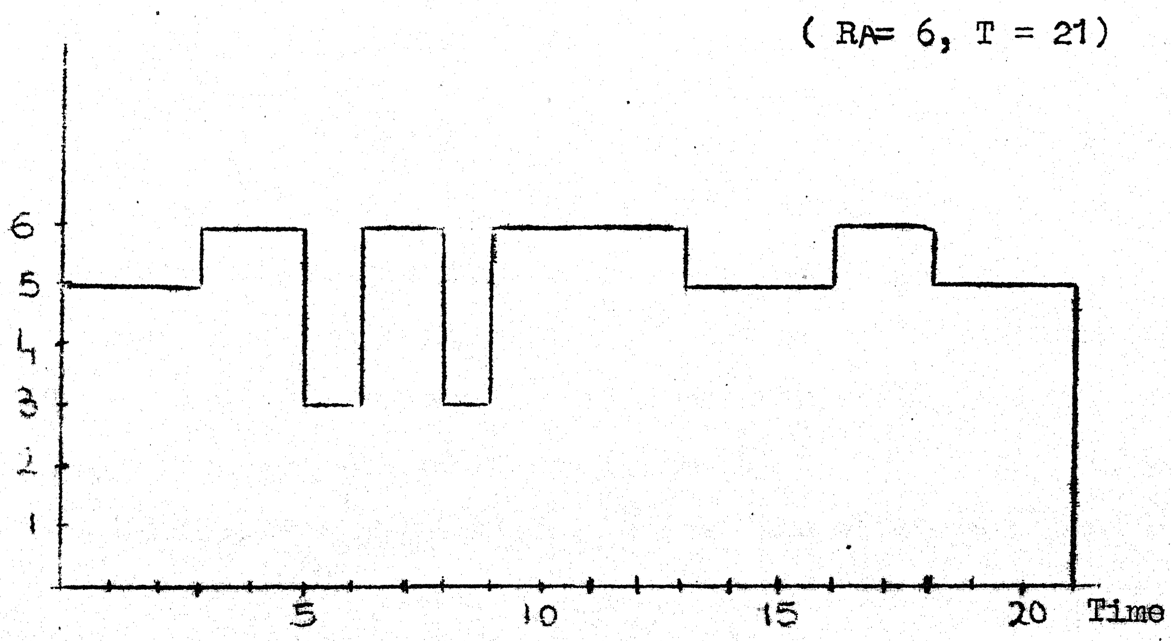


Resource Profile

Fig. 4.03



Scheduled graph



Resource Profile

Fig. 4.04

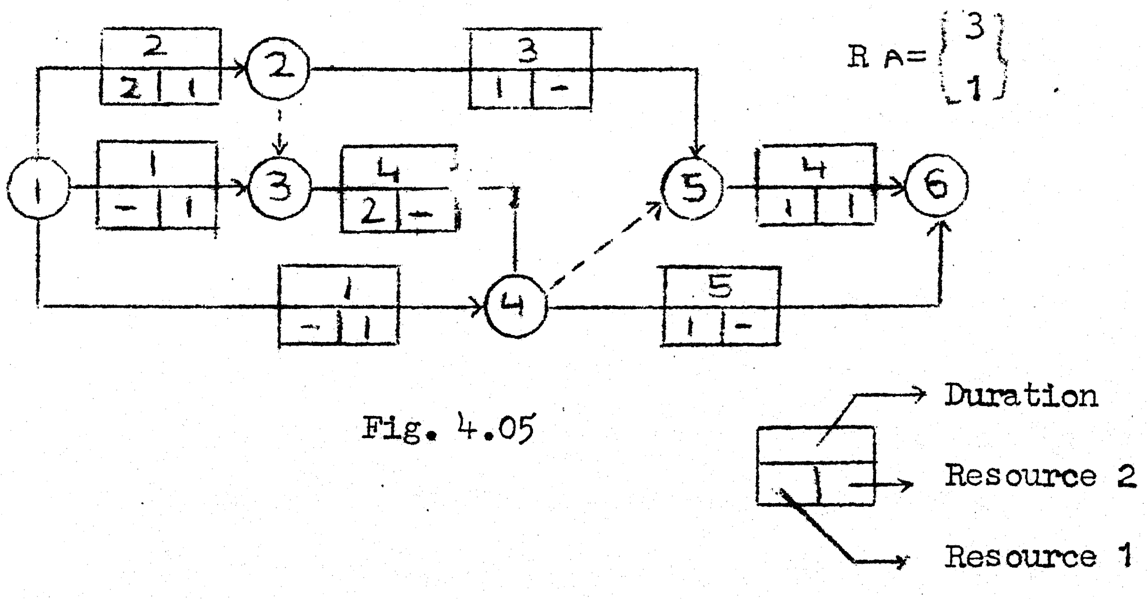


Fig. 4.05

<u>Network 2:</u>

A 9 activity, 2 resources, event-on-node network is given in Fig. 4.05. This problem illustrates the minimization of make-span and throughput times for the case of non-repetitive and repetitive projects respectively. For the resource availability of resource type 1 and 2 as 3 and 1 respectively, the scheduled graph and the resource profiles are given in Fig. 4.06. The make-span obtained is 12 units of time and the number of iterations required are 1270. When the project is repeated every 5 units of time, we get, for the resource availability of type 1 and 2 as 6 and 3 respectively, the scheduled graph and the resource profiles as shown in Fig. 4.07. The throughput time is 15 units and the number of iterations required are 373. The resource utilization of type 1 and 2 are 4/5 and 8/15. On decreasing the resource availability of type 2 to 2 (because its utilization is lower) we don't get any feasible solution for the maximum allowable throughput time of 15 units. Next, takingthe resource availability of type 1 and 2 as 5 and 3 respectively, we get the scheduled graph and resource profiles as shown in Fig. 4.08. The throughput time is 15 units and the number of iterations required are 1688. The resource utilizations for resource type 1 and 2 are 24/25 and 8/15 respectively.

Network 3:

A 2 jobs,3 machines flow-shop problem is given in Fig. 4.9. The total completion time of job 1 and job 2 are 14 and 12 units respectively. The penalties incurred for the non-completion of jobs in time t are as given below:

$P_{1,11} = 1$	$P_{1,13} = 4$	$P_{2,9} = 1$	$P_{2,11} = 1$
$P_{1,12} = 2$	$P_{1,14} = 5$	$P_{2,10} = 1$	$P_{2,12} = 3$

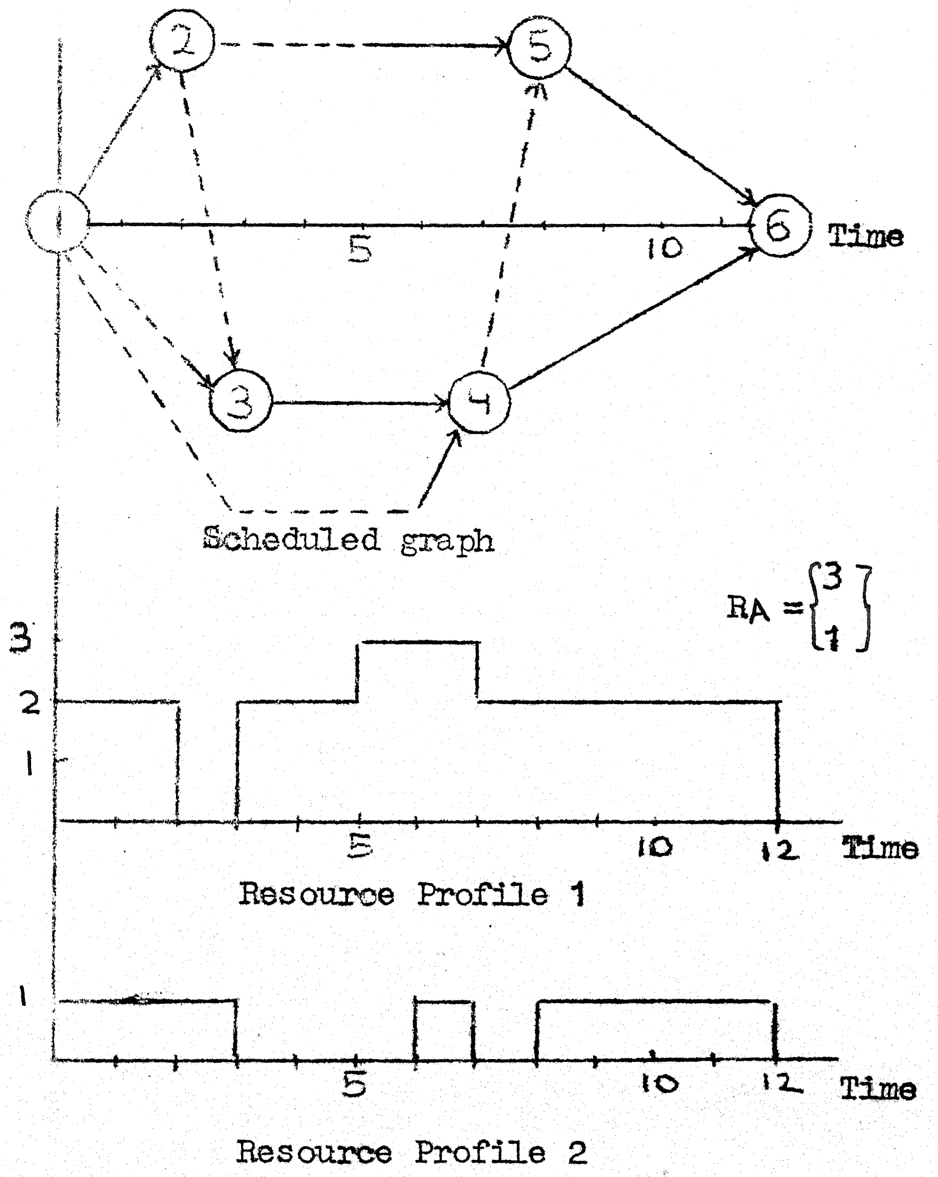
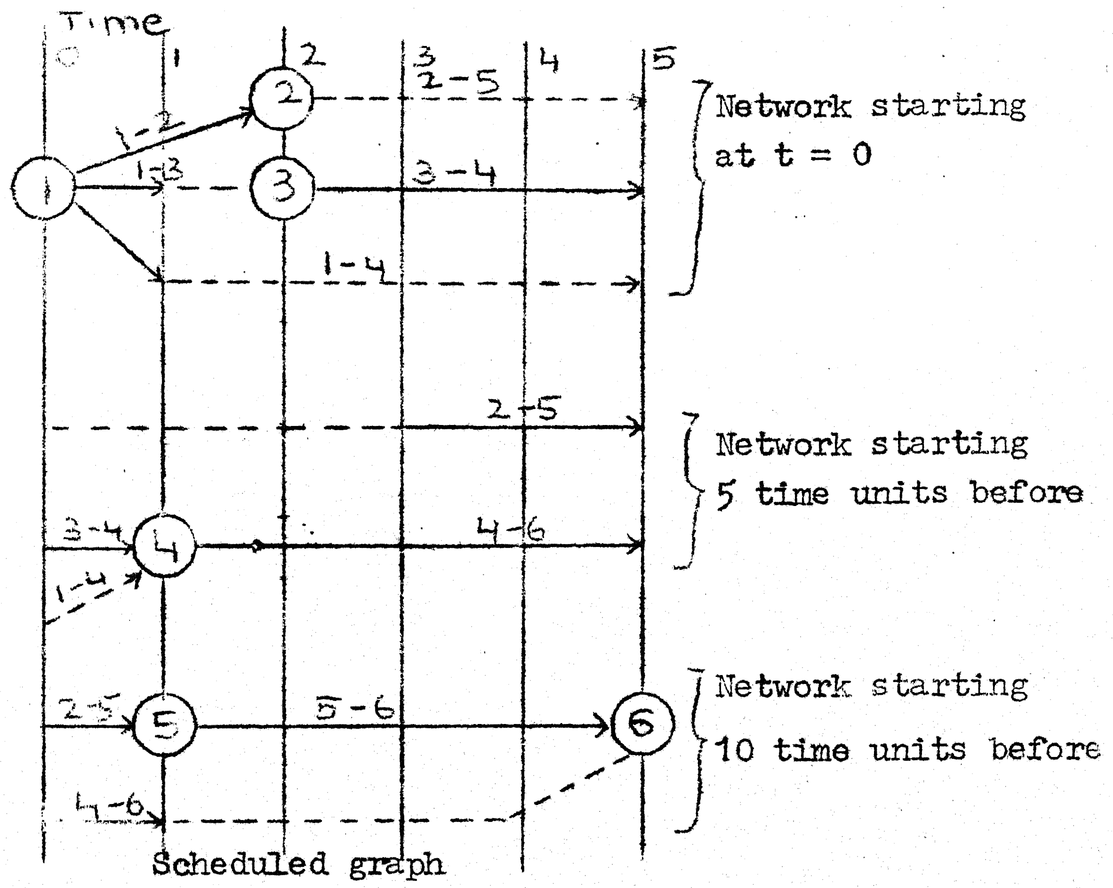


Fig. 4.06

Using Balas' algorithm for minimizing the total penalty we get the scheduled graph as shown in Fig. 4.10. The number of iterations required are 3159.


Time
0
1
2
3
4
5
1-2
1-3
1-4
2-5
3-4
4-6
5-6
1
2
3
4
5
6
Network starting at t = 0
Network starting 5 time units before
Network starting 10 time units before
Scheduled graph

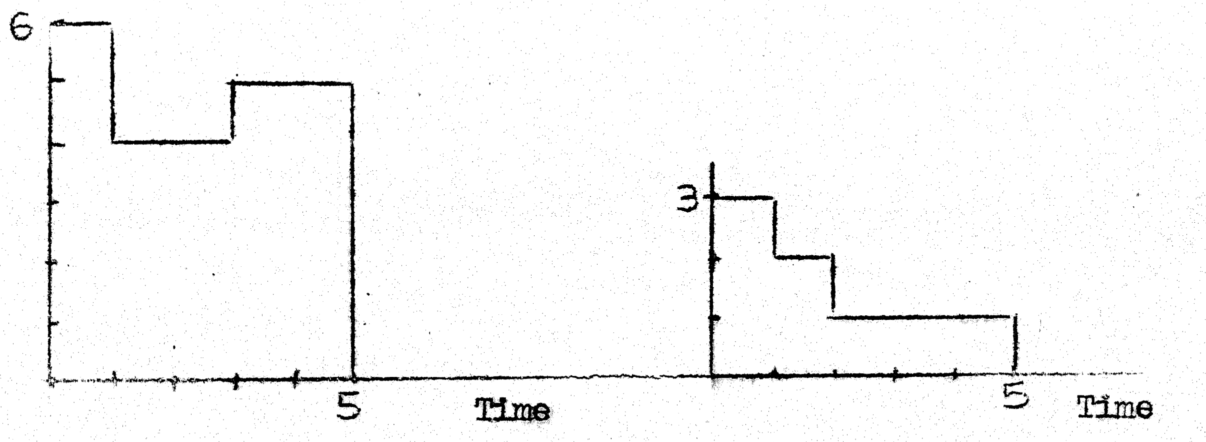
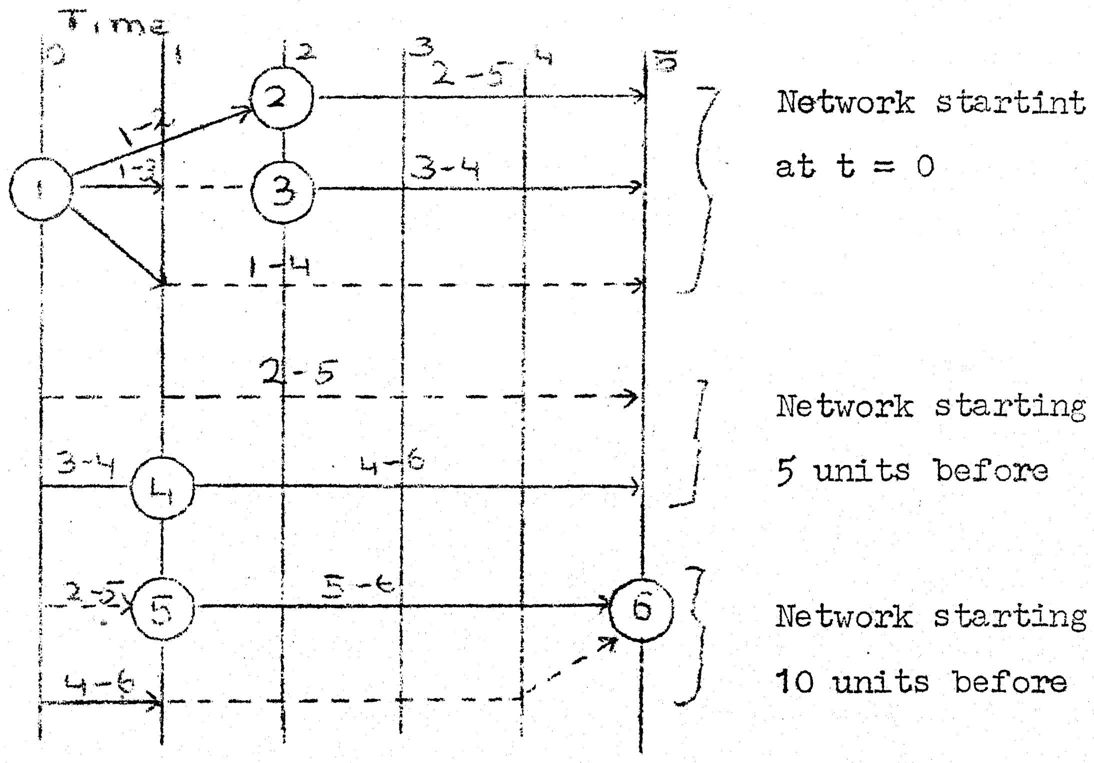
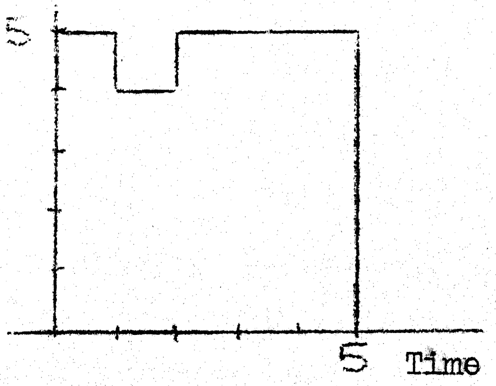

6
5
Time
Resource Profile 1
3
5
Time
Resource Profile 2

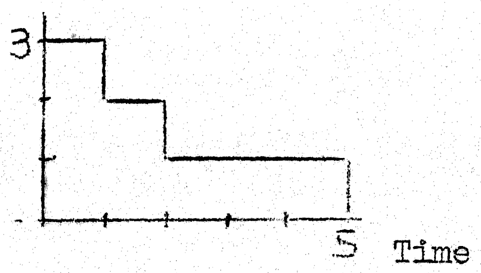
Fig. 4.07


Time
0
1
2
3
4
5
1-2
1-3
1-4
2-5
3-4
4-6
5-6
Network startint at t = 0
Network starting 5 units before
Network starting 10 units before

Scheduled graph



Resource Profile 1



Resource Profile 2

Fig. 4.08

Network 4:

A 3 jobs, 3 machines job-shop problem is given in Fig. 4.11. The sequences of the jobs on the machines are prespecified as follows:

Machine sequence for job 1 : (1, 3, 2)

Machine sequence for job 2 : (3, 2, 1)

Machine sequence for job 3 : (2, 1, 3)

Minimizing the make-span of the project we get the scheduled graph as shown in Fig. 4.12. The make-span is 13 units of time and the number of iterations required are 56. Both networks 3 and 4 assume availability of 1 unit of resource (machine) only.

Network 5:

A 22 activity, 1 resource event-on-node network is given in Fig. 4.13. For the project duration of 30 units, the attempt has been to achieve the maximum resource utilization. Decreasing the resource requirement from 9 units (corresponding to early start schedule) to 6 unit we obtain the scheduled graph and the resource profile as given in Fig. 4.14. The number of iterations required are 101.

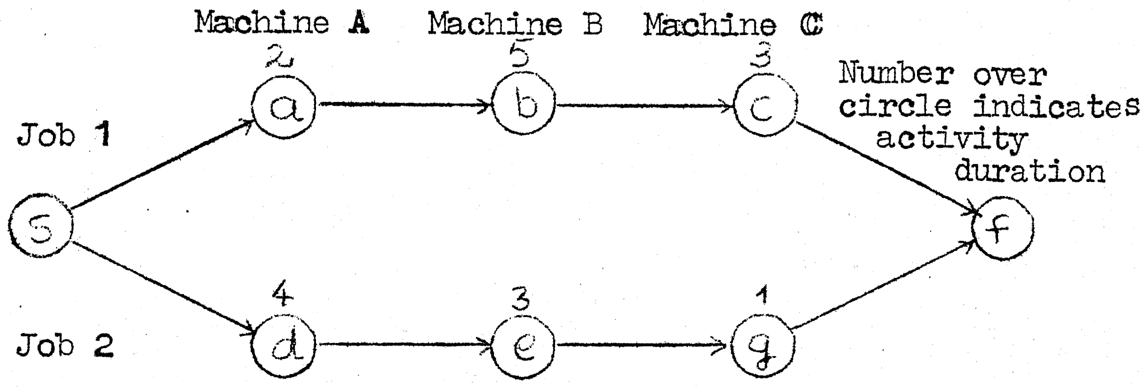


Fig. 4.09

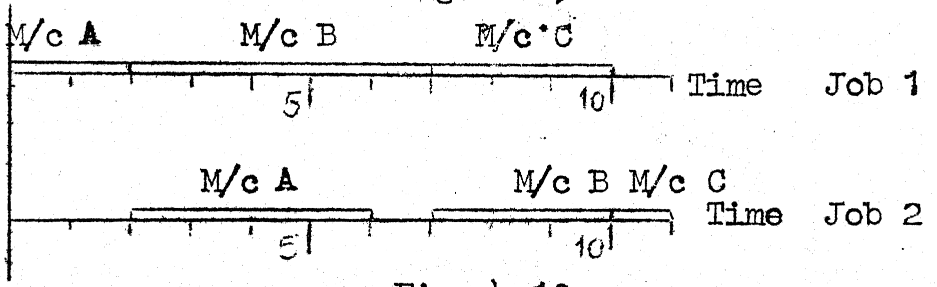


Fig. 4.10

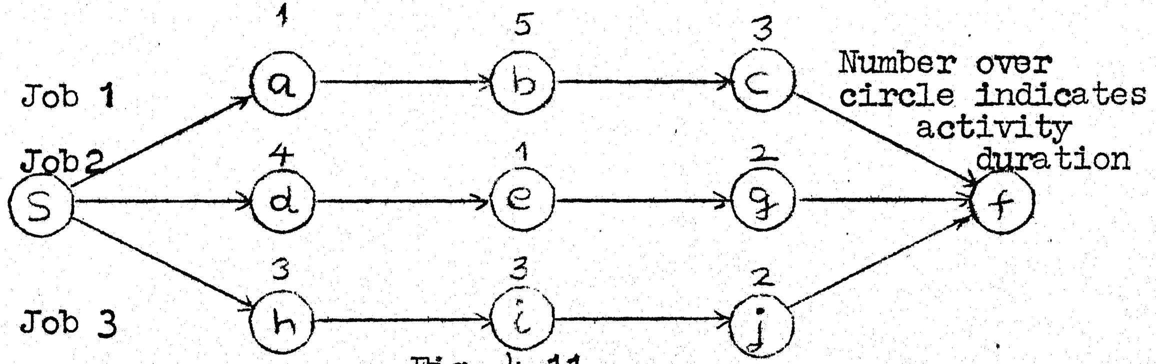


Fig. 4.11

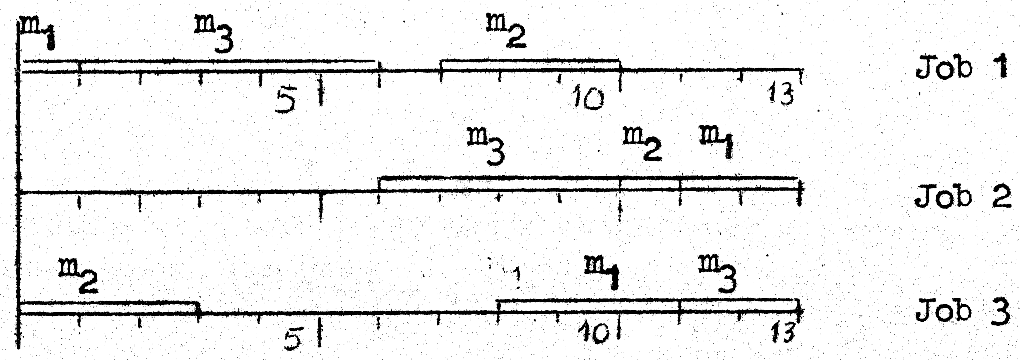


Fig. 4.12

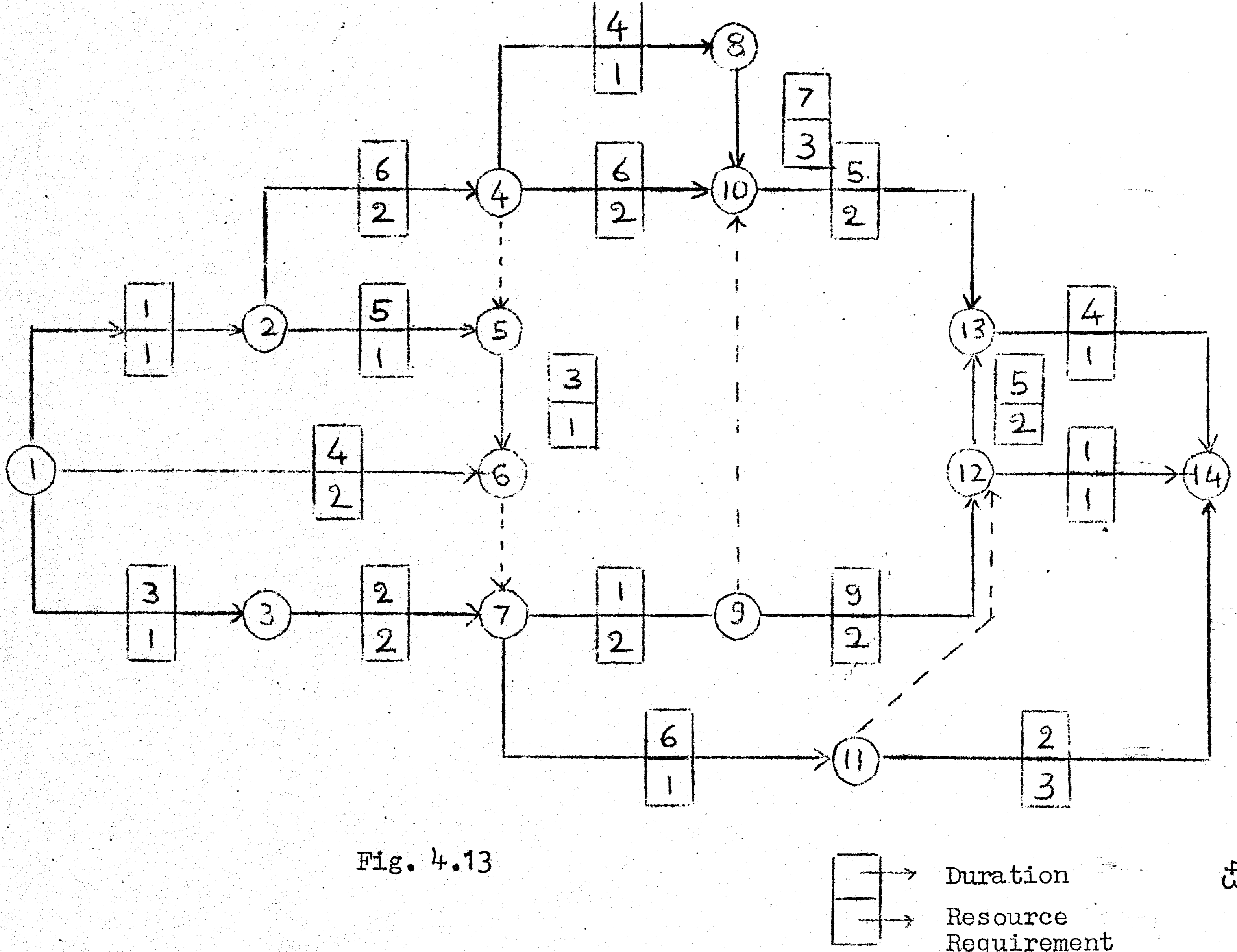
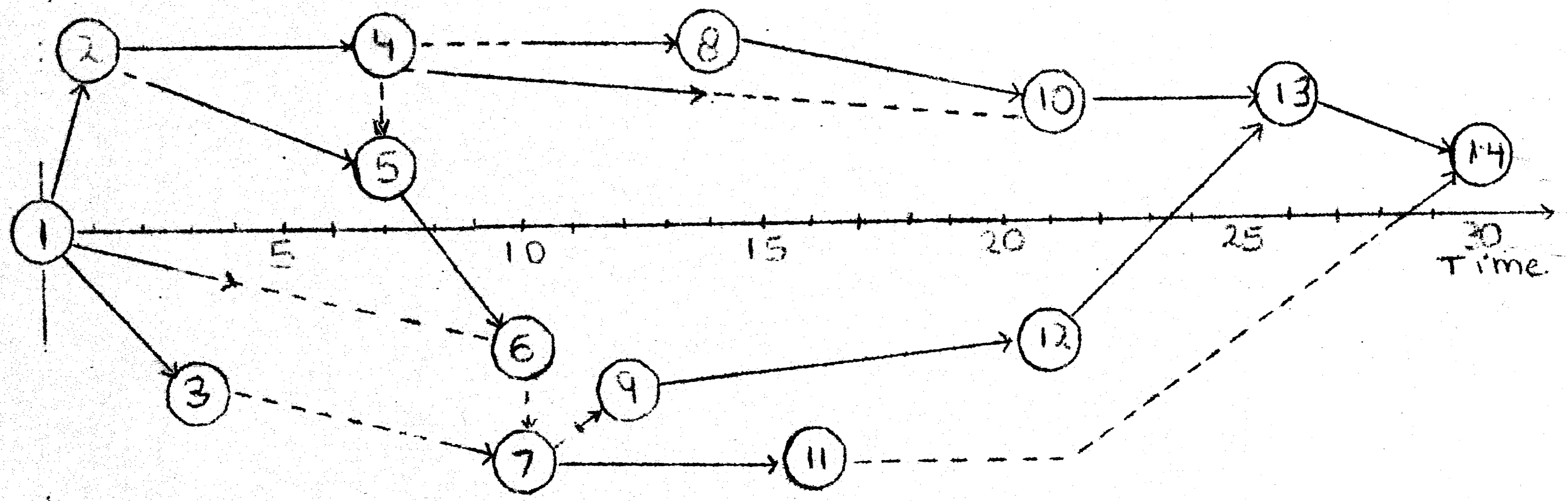
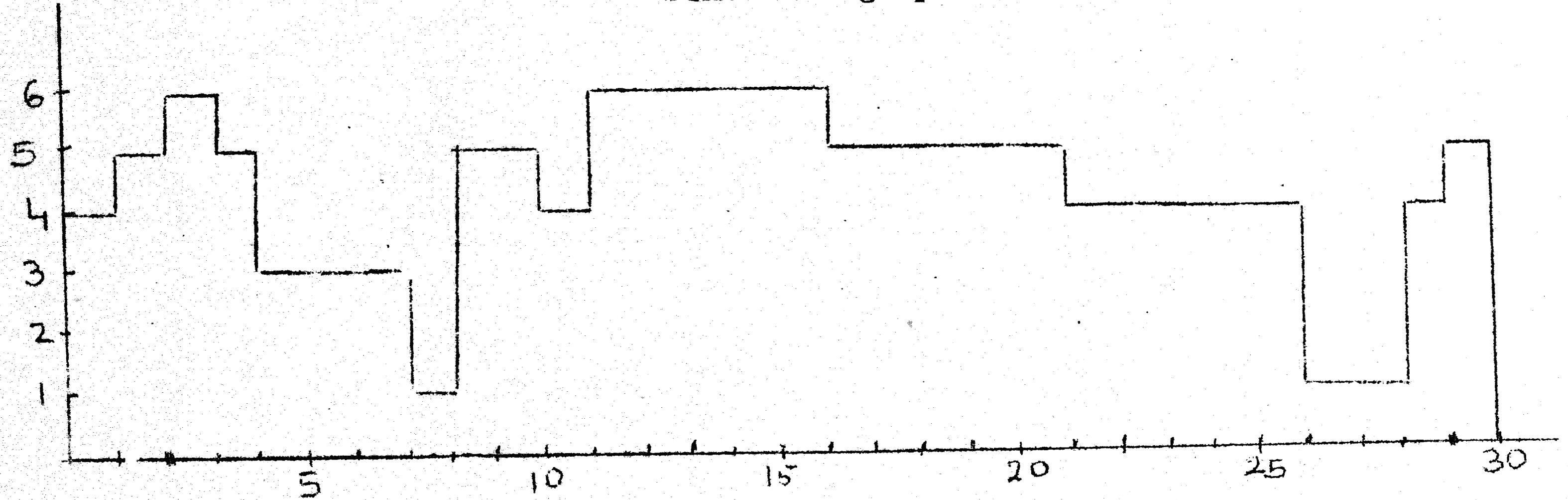


Fig. 4.13



Scheduled graph



Resource Profile

Fig. 4.14

<u>Network 6:</u>

An example of inter related projects requiring two resources is given in Fig. 4.15. Projects 1 and 2 consist of 7 and 9 activities. Activity 2 - 4 of project 1 has to precede activity 10 - 12 of project 2. For the resource profiles as shown inFig. 4.16. The number of iterations required are 5016.

<u>Network 7:</u>

A 16 activity, 1 resource network is given in Fig. 4.17. In this example activities 1 - 2, 3 - 8, 5 - 7 6 - 9 and 8 - 11 have to be carried out in the first department and the remaining in the other. The resource availability for the first department is 2 units while for the second department it is 3 units. The scheduled graph and the resource profile for the minimum make-span are given in Fig. 4.18. The number of iterationsrequired are 81.

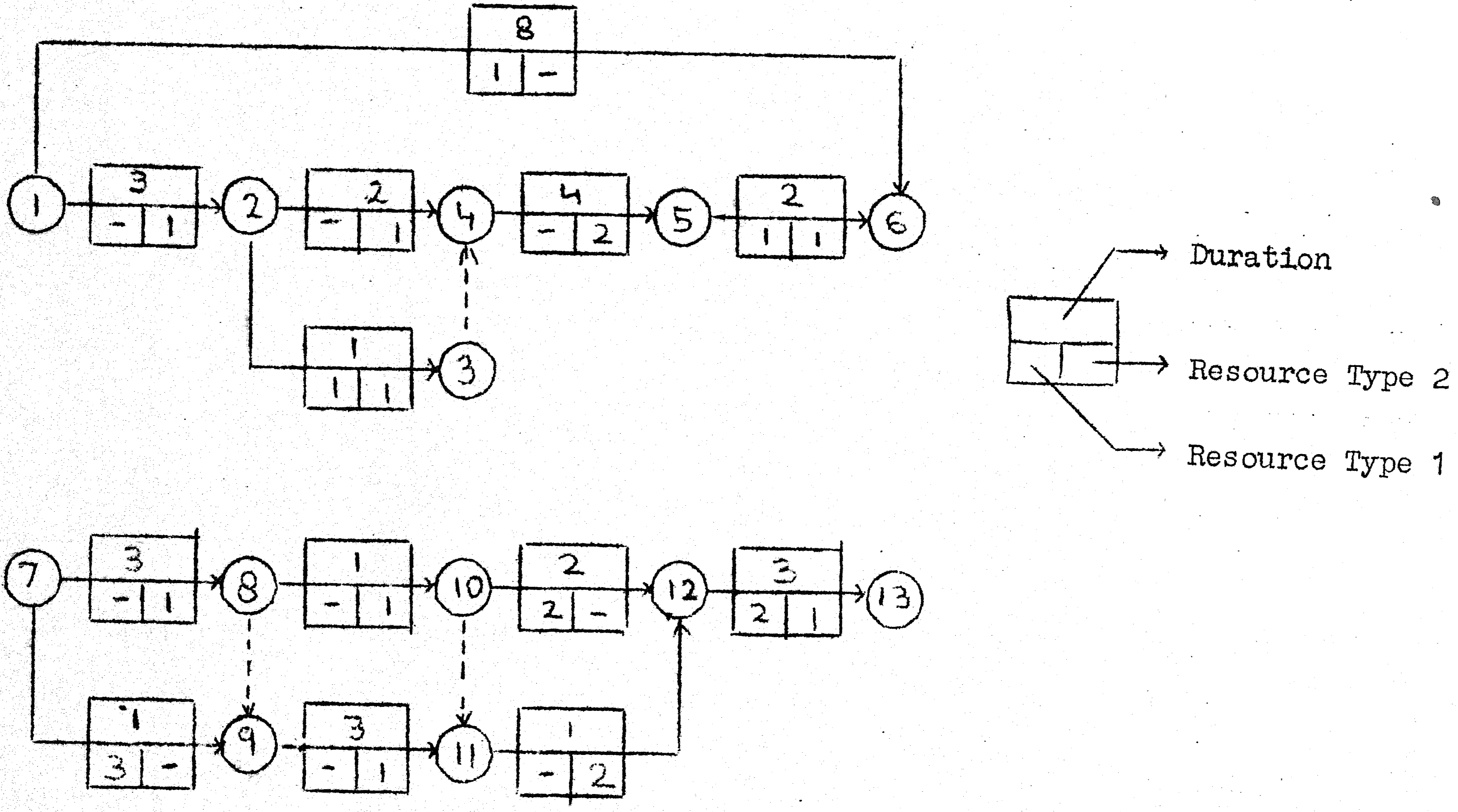
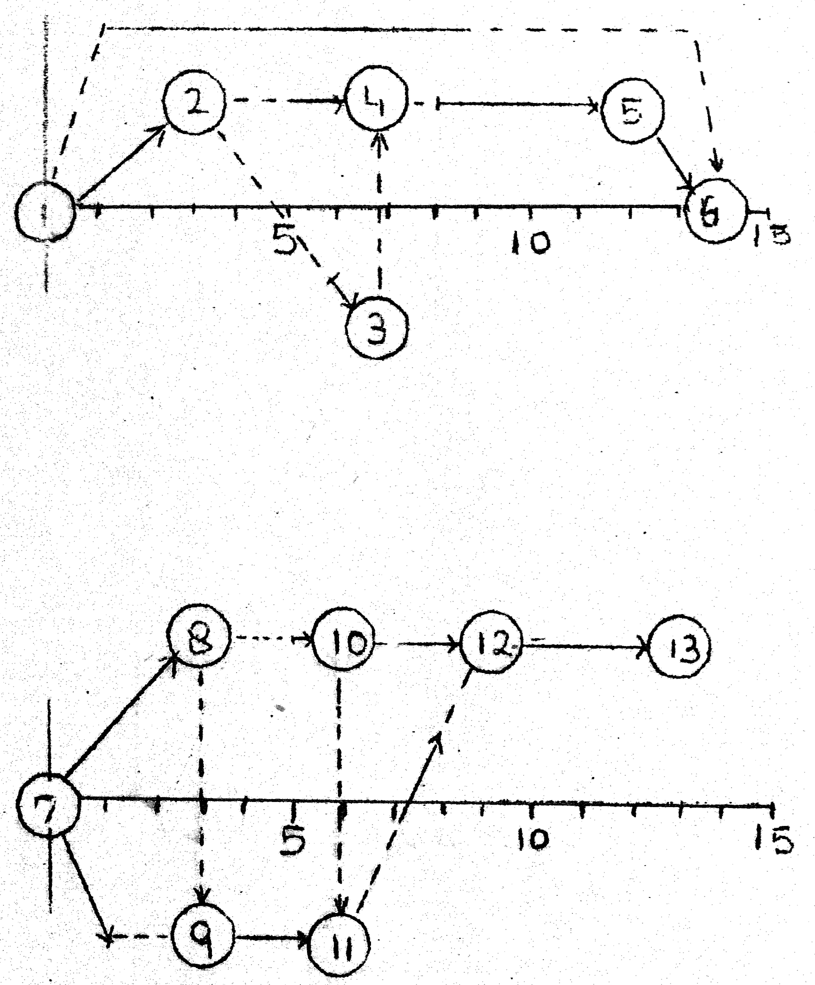
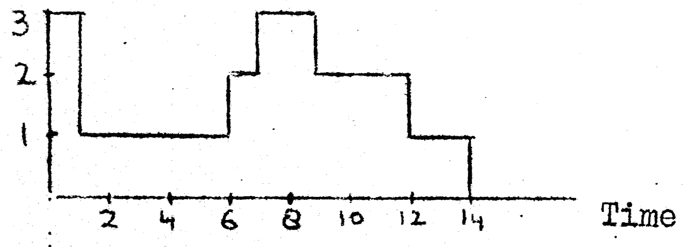


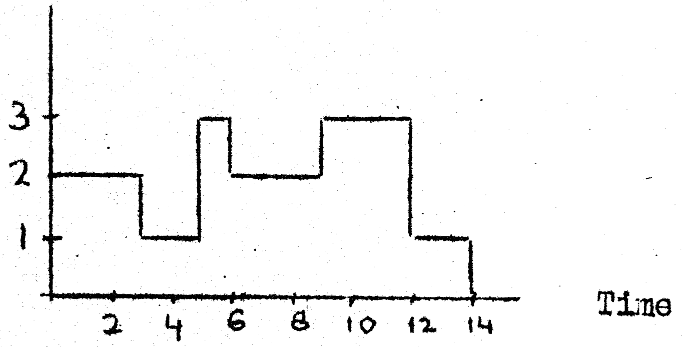
Fig. 4.15



Scheduled graph



Resource Profile 1



Resource Profile 2

Fig. 4.16

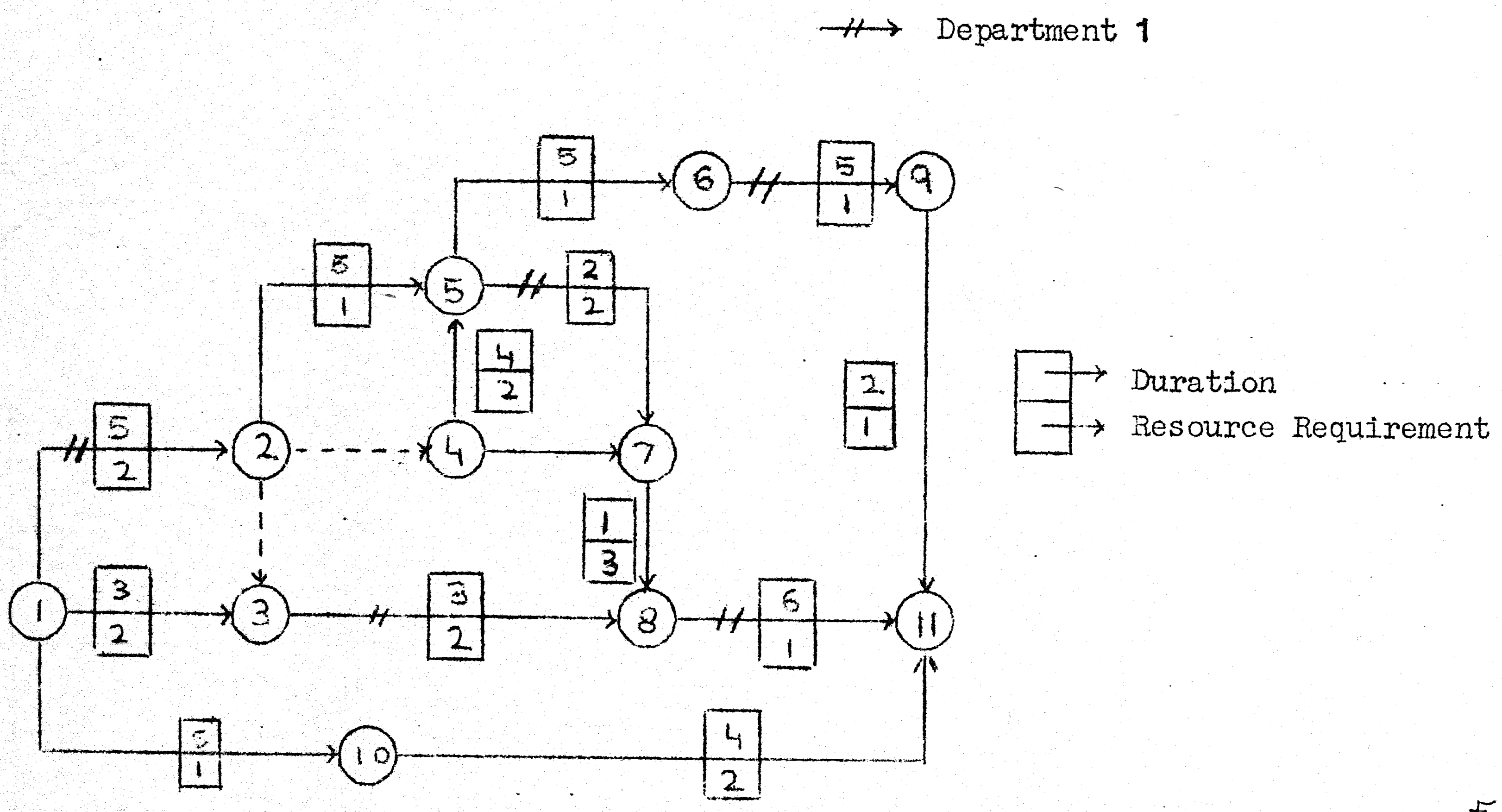
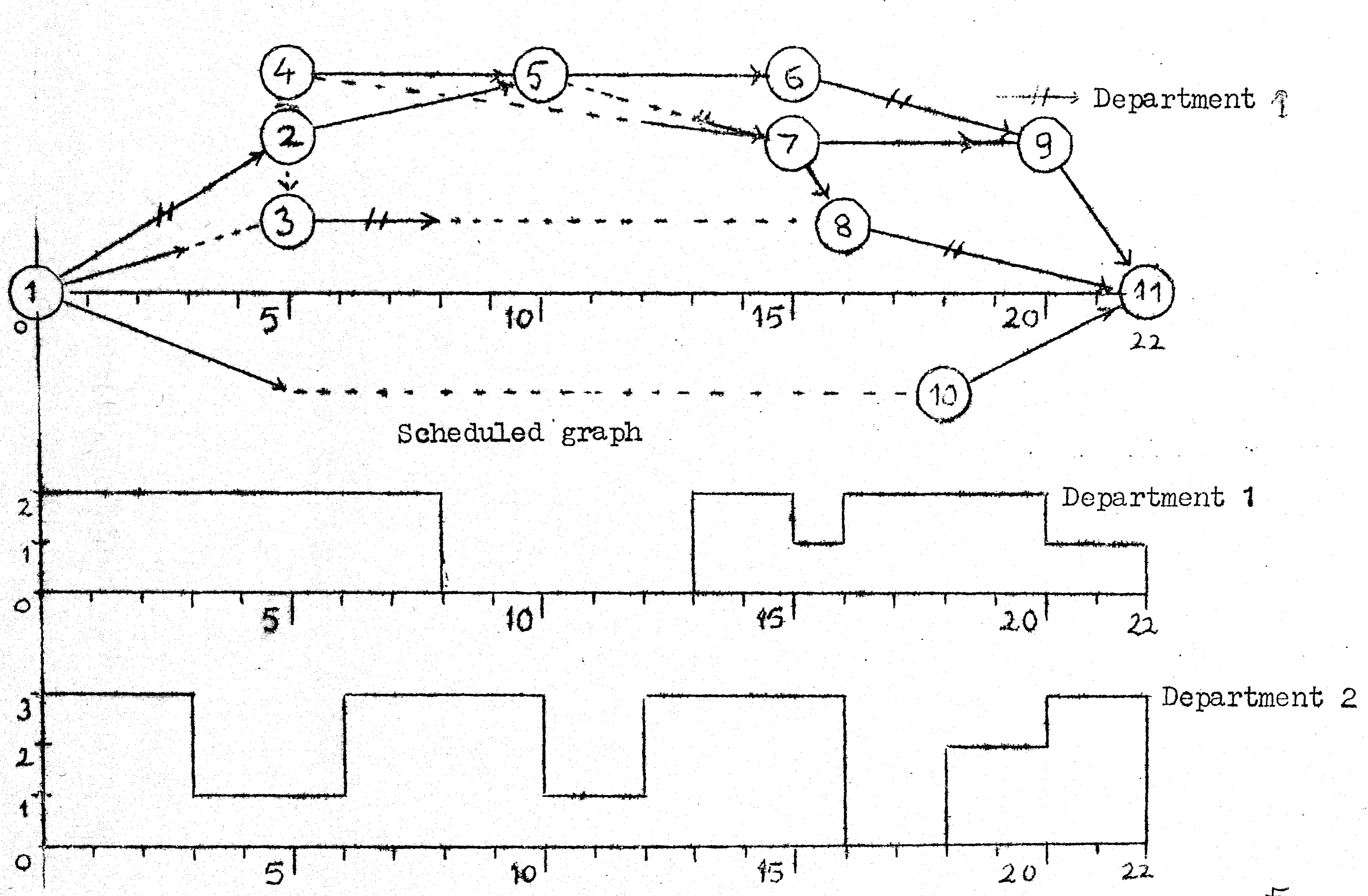

Department 1
Duration
Resource Requirement
1
2
3
4
5
6
7
8
9
10
11

Fig. 4.17



1
2
3
4
5
6
7
8
9
10
11
Department 1
0
5
10
15
20
22
Scheduled graph
Department 1
Department 2

Fig. 4.18

CHAPTER 5

INFERENCES AND SCOPE FOR FURTHER WORK

The following inferences have been drawn from the study of the generalized formulation and the solutions of the project scheduling problems.

1. The Balas' algorithm, used in this work, turns out to be quite efficient for finding the feasible solutions but it takes many iterations to obtain optimal from the generated feasible solution.

2. The proposed formulation requires $(\sum_{\forall i} \sum_{\forall j} (LA_{ij} - EA_{ij}) + \sum_{\forall i} (LP_i - EP_i))$ variables for the project scheduling problems for minimum penalty and throughput times. The number of variables, when the objective is to minimize make-span are $(\sum_{\forall i} \sum_{\forall j} (LA_{ij} - EA_{ij}) + (T - \max. EP_i))$. However, Bowman's (24) and, Moodie and Mandeville's formulations require $(I \cdot \sum_{\forall i} N_i \cdot T)$ variables. Thus, the proposed algorithm requires much less number of variables than required by either of the two algorithms. Therefore this algorithm can be used

to handle larger problems efficiently. However, as the number of activities increase beyond a certain limit the model becomes unmanageable. Furthermore it should be noted that when the duration of the project is increased by one unit of time, the number of variables increase by $(\sum_{i=1}^{I} N_i + I)$ for the objective criteria of minimum penalty and throughput time. In case of minimum make-span they increase by $(\sum_{i=1}^{I} N_i + 1)$.

3. The proposed formulation has an edge over Fischer's(28) formulation. This is because in the latter we have the assumption of availability of one unit of resource throughout the project duration. When this assumption is relaxed the efficiency of his method goes down while it is not so for the proposed algorithm. Furthermore, Fischer's model is useful when there are many projects with relatively less number of variables in each of them while the proposed formulation is preferable when the number of activities in a project are large and there are fewer projects. This is due to the fact that in Fischer's method the number of feasible schedules increase very

rapidly with the increase in number of activities in a project.

4. A comparison of the proposed algorithms with Davis and Heidorn's[19] algorithm indicates that the efficiency of the former does not go down for the large activity networks, as is the case with the latter.

5. The Resource - Time - Varying algorithm suggested in this dissertation has its relevance for the specialized 2 - dimensional cutting stock problems (e.g., wood; where the grain structure forces the sawing to be done in a special direction only). If the total length of the plate, total width of the plate, the length of the plates to be cut, the width of the plates to be cut and the order in which they have to be cut are to be designated by project span, resource availability, resource requirement, activity duration and sequencing constraints respectively, then the proposed algorithms give the minimum area of the plate.

6. As illustrated in Chapter IV, the generalized formulation proposed in this dissertation can

accommodate a variety of project scheduling problems.

cope for Further Work:

The proposed algorithm can be further improved by ncorporating the following features:

1. Activity completion and sequencing constraints of the proposed formulation possess a special structure. Furthermore, for all the projects (except for inter-related projects) the activity completion and sequencing constraints are independent while the project completion and resource requirement constraints are dependents. These facts can be used to develop efficient algorithms.

2. The flow-shop and the job-shop problems possess a special structure that, except for the first and last activities, all other activities have exactly one predecessor and one successor. Furthermore the resource availability is 1 unit throughout the project duration. These properties should be utilized in accomplishing faster computations.

3. The proposed formulation yield a very sparse matrix. Hence the properties of sparse matrices

should be used to achieve computational efficie-
ncy.

4. Certain modifications should be incorporated to take into account the resource substitution and the crashing of the activities.

BIBLIOGRAPHY

1. Martino, R.L., 'Resource Management', McGraw Hill Publications, 1969.

2. Wiest, J.D., 'A Heuristic Model for Scheduling Large Projects', Management Science, Vol. 13, No.6, 1967, pp. B359 - B377.

3. Wiest, J.D., 'Some Properties of Schedules for Large Projects with Limited Resources', Operations Research, Vol. 12, No. 3, 1964, pp. 395-418.

4. Burgess, A.R. and J.B. Killebrew, 'Variation in Activity Level on a Cyclical Arrow Diagram', Journal of Industrial Engineering, Vol. 13, No. 2, 1962, pp. 226-230.

5. Wilson, R.C., 'Assembly Line Balancing and Resource Levelling', University of Michigan Summer Conference, Production and Inventory Control, 1964.

6. Charnes, A., and W.W. Cooper, 'A. Network Interpretation and Directed Subdual for Critical Path Scheduling', Journal of Industrial Engineering, Vol. 13, No. 4, 1962, pp. 213-218.

7. Dewitte, L., 'Manpower Levelling of PERT Networks', Data Processing for Science/Engineering, March-April, 1964.

8. Levy, F.K., G.L. Thompson and J.D. Wiest, 'Multiship, Multishop, Workload Smoothing Program', Naval Research Logistics Quarterly, Vol. 9, No. 1, 1962, pp. 37-44.

9. Kelley, J.E., Jr., 'The Critical Path Method: Resource Planning and Scheduling' Factory Scheduling Conference, Carnegie Institute of Technology, May 10-12, 1962.

10. Lambourn, S., 'RAMPS-A New Tool in Planning and Control', The Computer Journal, Vol. 6, No. 1, 1963, pp. 33-37.

11. Bennington, G.E., and L.F. McGinnis, 'A Critique of Project Planning with Constrained Resources', paper presented in a Symposium in Raleigh, N.C.

12. Florian, M.P. Trepant and G. McMohan, 'An Implicit Algorithm for the Machine Sequencing Problem, Management Science, Vol. 17, No. 12, 1971, pp. B782-B792.

13. Schrage, L., 'Solving Resource Constrained Problem by Implicit Enumeration Non-Preemptive Case', Operations Research, Vol. 18, No. 2, 1970, pp. 263-278.

14. Balas, E., 'Machine Scheduling via Disjunctive Graphs - An Implicit Enumeration Algorithm', Operations Research, Vol. 17, No. 6, 1969, pp. 941-957.

15. Gorenstein, S., 'An Algorithm for Project Scheduling with Resource Constraints', Operations Research, Vol. 20, No. 4, 1972, pp. 835-850.

16. Sussman, B., 'Scheduling Problems with Interval Disjunctions', Technical Report No.70-5, Operations Research House, Stanford University (1970).

17. Schrage, L., 'Solving Resource Constrained Network Problems by Implicit Enumeration - Preemptive Case', Operations Research, Vol. 20, No. 3, 1972, pp.668-677.

18. Mason, A., and C.L., Moodie, 'A Branch and Bound Algorithm for Minimizing Cost in Project Scheduling', Management Science, Vol. 18, No. 4, 1971, pp. B158-B173.

19. Davis, E.W., and G.E. Heidorn, 'An Algorithm for Optimal Project Scheduling under Multiple Resource Constraints', Management Science, Vol. 17, No. 12, 1971, pp. B803-B816.

20. Johnson, T.J.R., '<u>An Algorithm for the Resource Constrained Project Scheduling Problems</u>', Ph.D. Thesis, M.I.T., 1967.

21. Moder, J.J. and C.R. Phillips, '<u>Project Management with CPM & PERT</u>', Reinhold Corporation, New York, 1964.

22. Greenberg, H.H., 'A Branch and Bound Solution to the General Scheduling Problems', <u>Operations Research</u>, Vol. 16, No. 2, 1968, pp.353-361.

23. Moodie, C.L., and D.E. Mandeville, 'Project Resource Balancing by Assembly Line Blanacing Techniques,' <u>Journal of Industrial Engineering</u>, Vol. 17, No. 7, 1966, pp. 377-381.

24. Bowman, E.H., 'The Schedule Sequencing Problem', <u>Operations Research</u>, Vol. 7, No. 5, 1959, pp. 621-624.

25. Manne, A.S., 'On the Job-shop Scheduling Problem', <u>Operations Research</u>, Vol. 8, No. 2, 1960, pp. 219-223.

26. Wagner, H.M., 'An Integer Linear-Programming Model for Machine Sequencing', <u>Naval Research Logistics Quarterly</u>, Vol. 6, No. 2, 1959, pp. 131-139.

27. Bowman, E.H., 'Assembly Line Balancing by Linear Programming', <u>Operations Research</u>, Vol. 8, No. 3, 1960, pp. 385-389.

28. Fischer, M.L., 'Optimal Solution of Scheduling Problems using Lagrange Multipliers: Part I', <u>Operations Research</u>, Vol. 21, No. 5, 1973, pp. 1114-1127.

29. Fischer, M.L., 'Optimal Solution of Scheduling Problems using Lagrange Multipliers: Part II', paper presented at <u>Symposium on the Theory of Scheduling and its Applications</u>, N.C., May 15-17, 1972.

30. Patterson, J.H. and W.D. Huber, 'A Horizon Varying Zero-One Approach to Project Scheduling', Management Science, Vol. 20, No. 6, 1974, pp. 990-998.

31. Gutjahar, A.L., and G.L.Nemhauser, 'An Algorithm for the Line Balancing Problem', Vol.11, No. 2, 1964, pp. 308-315.

32. 'Symposium on the Theory of Scheduling and its Applications', Edited by S.E. Elmaghraby, Springer, Verlag, Berlin, Heidelberg, New York.

33. Davis, E. W., 'Resource Allocation in Project Network Models - A Survey', Journal of Industrial Engineering, Vol. 17, No. 3, 1966, pp. 177-188.

APPENDIX A

BOWMAN'S SCHEDULE SEQUENCING FORMULATION

Consider the case of three products X, Y, Z which are to be processed on four machines A, B, C, D. The sequences for products X,Y,Z are $A \rightarrow B \rightarrow C \rightarrow D$, $C \rightarrow A \rightarrow D \rightarrow B$ and $D \rightarrow A$ respectively. The project is to be completed by time T. Let $X_{A:t} = 1$ denote that product X is being processed on machine A at time t and let $X_{A:t} = 0$ otherwise. The various constraints are formulated as given below:

1. Job Completion Constraints:

If d_{XA} denotes the duration of the product X on machine A, then the job completion constraint is expressed by the relationship:

$$\sum_{i=1}^{T} X_{A:i} = d_{XA} \quad \text{for all products and machines.} \qquad (1)$$

2. Non-interference Constraints:

This implies that two or more products cannot be handled simaltaneously on a machine. Expressing this mathematically we get,

$X_{A:t} + Y_{A:t} + Z_{A:t} \leq 1$ for $t = 1,\ldots,T$ and all machines.

3. Precedence Constraints:

To ensure that product X is processed first on machine A and then on B we must satisfy the following inequality.

$$d_{XA}\, X_{B:j} \leq \sum_{i=1}^{j-1} X_{A:i} \quad \text{for } j = 1,\ldots,T \text{ and all sequences,}$$

For products Y and Z the precedence relationships can be expressed similarly.

4. Non-interruption Constraints:

The jobs, once started, should not be interrupted. Mathematically this is expressed as,

$$d_{XA}\, X_{A:i} - d_{XA}\, X_{A:i+1} + \sum_{j=i+2}^{T} X_{A:j} \leq 5$$

for $j = i,\ldots,T$ and all jobs.

Objective Function:

For minimizing the makespan of the project the following objective function is used.

Min. O.F. = $1(X_{D:23} + Y_{B:23} + Z_{A23}) + 4(X_{D:24} + Y_{B:24} + Z_{A:24}) + 16(X_{D:25} + Y_{B:25} + Z_{A:25}) + \ldots + K_T(X_{D:T} + Y_{B:T} + Z_{A:T})$.

APPENDIX B

MODDIE AND MANDEVILLE'S FORMULATION FOR PROJECT RESOURCE BALANCING

Moodie and Mandeville have used the following notations for resource balancing problem.

Let A_i denote resource level of activity i on day A; Υ_i resource level of activity i on any day; β_i duration of activity i in days and $\emptyset_A$ maximum resource available. The various constraints are expressed as follows:

1. Activity Completion Constraints:

Each activity must be completed in the specified duration. To ensure this we require the following relationships:

$$A_1 + B_1 + C_1 + \ldots + G_1 = \beta_1 \Upsilon_1$$

$$A_2 + B_2 + C_2 + \ldots + G_2 = \beta_2 \Upsilon_2$$

2. Resource Constraints:

Each activity should be performed at the specified level of resource requirement. Mathematically it is expressed as,

$$\frac{1}{\tau_1} A_1 + I_{A_1} = 1 , \frac{1}{\tau_1} B_1 + I_{B_1} = 1, \frac{1}{\tau_2} A_2 + I_{A_2} = 1 \ldots$$

where I's are either 0 or 1.

3. Precedence Constraints:

The precedence relationship can be represented as follows:

$$\frac{1}{\tau_1} B_2 \leqslant \frac{1}{\tau_1 \beta_1} A_1$$

$$\frac{1}{\tau_1} C_2 \leqslant \frac{1}{\tau_1 \beta_1} A_1 + \frac{1}{\tau_1 \beta_1} B_1 \quad \ldots$$

4. Resource Balancing Constraints:

To balance the resource level over the total projects we must have,

$$A_1 + A_2 + \ldots + A_8 \simeq 0$$

$$B_1 + B_2 + \ldots + B_8 \simeq A_1 + A_2 + \ldots + A_8$$

$$C_1 + C_2 + \ldots + C_8 \simeq B_1 + B_2 + \ldots + B_8$$

$$\vdots$$

$$0 \simeq N_1 + N_2 + \ldots + N_8$$

5. Non-interruption Constraints:

An activity once started should be completed without interruption. To ensure this we have,

$$(\beta_1 - 1) A_1 \leq B_1 \quad ; \quad (\beta_1 - 1) B_1 \leq C_1 + A$$

Objective Function:

The objective function used by him is concerned with the minimization of cost of applying the resources.

APPENDIX C

FISCHER'S FORMULATION FOR OPTIMAL SCHEDULING USING LAGRANGE MULTIPLIERS

Fischer has developed a zero-one formulation for the project scheduling problems using Lagrange Multipliers. Various notations used by him are,

i	: number of projects; $i = 1,\ldots,I$
t	: time period ; $t = 1,\ldots, T$
k	: types of resources; $k = 1,\ldots,K$
n_i	: number of activities on project i
p_{ij}	: duration of activity ij
t_{ij}	: start time of activity ij
f_{ij}	: finish time of activity ij
c_{iq}	: cost associated with qth job-schedule of project i
a_{iqkt}	: the amount of resource type k required at time t for qth schedule of project i.

The various constraints are expressed as follows :

1. Non-preemption Constraints : An activity, once started, should not be interrupted. This can be expressed as follows :

$$f_{ij} = t_{ij} + p_{ij} \qquad \ldots \qquad (1)$$

2. <u>Precedence Constraints</u>:

If an activity ij follows the activity il, it must satisfy the following inequality.

$$t_{ij} \geqslant f_{il} = t_{il} + p_{il} \qquad (2)$$

3. <u>Project Completion Constraints:</u>

If $f_i = \max\limits_{j=1,\ldots,n_i} (f_{ij})$, then we have

$$f_i \leqslant T \qquad \ldots \qquad (3)$$

Let a job schedule for the project i be any schedule that satisfies the constraints (1), (2) and (3).

Let $X_{iq} = 1$ indicate that we use the qth schedule for project i.

$= 0$, otherwise.

Then we have,

$$\left.\begin{array}{ll} \text{Minimize} & \sum\limits_{iq} C_{iq} X_{iq} \\ \text{s.t.} & \sum\limits_{q} X_{iq} = 1 \quad i = 1,\ldots,I \\ \multicolumn{2}{l}{\sum\limits_{iq} \alpha_{iqkt} X_{iq} \leqslant R_{kt} \quad t = 1,\ldots,T,\ k = 1,\ldots,K} \\ \multicolumn{2}{l}{X_{iq} = 0 \text{ or } 1 \quad q = 1,\ldots Q_i ,\ i = 1,\ldots,I} \end{array}\right\} \text{(A)}$$

The above problem can be converted into the following Lagrangian problems.

$$\left.\begin{array}{ll} W(U) = \underset{X_{iq}}{\text{Min.}} \left(\sum_{X_{iq}} C_{iq} X_{iq} + \sum_{kt} U_{kt} \left(\sum \alpha_{iqkt} - R_{kt} \right) \right) & \\ \text{s.t.} \quad \sum_{q} X_{iq} = 1 & i = 1,\ldots,I \\ \qquad X_{iq} = 0 \text{ or } 1 & q = 1,\ldots,Q_i \\ & i = 1,\ldots,I \end{array}\right\} \quad (B)$$

Now, let $\pi_{iq}(U) = C_{iq} + \sum_{kt} \alpha_{iqkt} U_{kt}$ and

$\pi_i (U) = \underset{q}{\text{Min.}}\ \pi_{iq} (U)$.

We can show that the dual of (A) with integrality relaxed is,

$$\left.\begin{array}{lll} \text{Maximize} \quad W = \sum_{i} \pi_i - \sum_{kt} R_{kt} U_{kt} & & \\ \text{s.t.} \quad \pi_i \leq C_{iq} + \sum_{kt} \alpha_{iqkt} U_{kt} & & q=1,..,Q_i \\ & & i=1,\ldots,I \\ \qquad U_{kt} \geq 0 & k = 1,..,K & \\ & t = 1,..,T & \end{array}\right\} \quad (C)$$

Where π_i, $i = 1,\ldots I$ are unrestricted in sign.

U_{kt} : Lagrange Multipliers are positive.

Even though (C) has numerous constraints, most of them are non-binding at a particular extreme point solution. This fact is used in developing a branch and bound algorithm in which we solve (C) by simplex method dealing with only a subset of binding constraints.

ME-1975-M-BAH-PRO